NOTE TO USERS

APPLICATION OF THE

DESIGN STRUCTURE MATRIX METHODOLOGY TO

THE DESIGN OF A MOTORIZED WHEEL

XIU YU SUN

A Thesis
in
The Department
of
Mechanical and Industrial Engineering

Presented in Partial Fulfillment of Requirements
for the Degree of Master of Applied Science (Mechanical Engineering) at
Concordia University
Montreal, Quebec, Canada

December, 2004

Your file Votre référence
ISBN: 0-494-04431-4
Our file Notre référence
ISBN: 0-494-04431-4

Abstract

Application of the Design Structure Matrix Methodology to the Design of a Motorized Wheel

Xiu Yu Sun

Effective engineering management tools are imperative in today's competitive environment. The Design Structure Matrix (DSM) is a tool that is used to plan and organize the product development process efficiently. The DSM is based on information flows between activities in the development process. All design tasks and their relationships are expressed in matrix form. Through algorithms designed to reorganize the matrix, an efficient execution order of development activities can be proposed. In addition, the DSM has a numerical function that evaluates the design process quantitatively in order to estimate the total time required to complete a project. In this thesis, three design models are presented in the application of the DSM in the product development process of an actual product: a motorized wheel. The detailed process of applying the DSM method in the product development process will be presented.

ACKNOWLEDGMENTS

I would sincerely like to thank my supervisor, Dr. Nadia Bhuiyan, for giving me the opportunity to pursue an insightful research topic. In the past two years, Dr. Bhuiyan has taught me a great deal about my academic life in Concordia University. Her intelligent guidance will be helpful in my future career. I also appreciate very much Dr. Linghua Kong's strong support in the design of the motorized wheel and the electrical wheelchair. His suggestions were meaningful for my design work.

Thanks also to my wife, Lina, and my son, Sebastian. They give me a happy family life.

This thesis is one of a series of projects that are being studied under the research theme, 'Application of DSM Methodology in Rehabilitation Engineering, Health Care and Clinic Science', under the supervision of Dr. Bhuiyan and Dr. Kong, President of Green BioMedi Technologies and Instruments and Research Fellow in the Center for Assistive Technology and Environmental Access (CATEA) at the Georgia Institute of Technology in Atlanta.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1 INTRODUCTION

1.1 New Product Development

Competition among firms today is intense, forcing companies to improve current products and create innovative new products in record times. Many companies, automotive for example, strive to launch new models on the market even though old models are still popular. Companies must pay more attention to the product development process in order to minimize development time and improve product quality. Appropriate methods to organize the product development process are required in order to do so effectively.

New product development (NPD) is the process of bringing a new product idea to market (Crawford, 1994). It can be viewed as a series of decision-making processes that are carried out by transforming information inputs into information outputs (Clark and Fujimoto, 1991; Galbraith, 1973). Such a process generally includes the following stages: product conception, product evaluation, product planning, product design, prototyping, manufacturing, and marketing. In the product conception stage, ideas for potential new products to be developed are generated based on information from the market and various other sources. In the product evaluation stage, the technical and economic feasibility of ideas are screened. Next, in the product planning stage, concrete plans to launch the development of feasible product ideas are scheduled. The next three stages, product design, prototyping, and manufacturing, involve the technical implementation of new product development and the final phase involves launching the product onto the marketplace. These stages are often carried out concurrently in order to reduce

development time. By adopting the information processing view of NPD, process improvement can be achieved by focusing on the study of the information flows needed to complete the process.

1.2 Complexity and Challenge in Product Design

Product design is a very important stage in NPD, and it involves cognitive and organizational processes by which artifacts are created (Ulrich 2001). An artifact can be a simple product, such as a gear, or a complex assembly, such as a transmission box. For example, in the design of a gear, not only should the geometric dimensions of the gear be considered, such as the module, teeth, and press angle, but also the material, strength, and toughness are important considerations. These items are the product parameters that decide the nature of a product.

A successful product design is the result of three key characteristics: reduced time, low design cost, and high product quality (Ulrich and Eppinger, 2004). A reduced design time highly influences the competitiveness of a company since it can more quickly send its products to market and reap benefits earlier from product development efforts. Shorter design time is also related to low design cost. Typically, design cost includes many components, such as labor, materials, technology, utilities, and so on. Therefore, the earlier a design process is completed, the more cost design teams can save. If design cost is too high, the products will be offered to customers at a high price and/or will not attract a high portion of the market. Consequently, these products will not be competitive, and the company also loses in market profile. High product quality is expected by customers, but the pursuit of high quality often leads to high design cost. Thus, it is very important to

balance the cost and quality to meet the needs of customers and achieve the desired profits.

How to realize a successful design is challenging for companies since the product design process is very complex: it is the process of gathering and transforming information and making decisions in order to find solutions to determine product parameters. For example, in choosing the material of a gear, decisions should meet the criteria of surface hardness, interior hardness, and heat treatment technology. The process of defining a product parameter can be called a design task. Since it involves the work of designing and choosing hundreds of parts, the design process is composed of many tasks. Even a single part, such as a gear itself includes numerous parameters. Once these tasks are completed, the design process is also completed. Design decisions such as these are critical since, once they are made, they can affect other organizational decisions (Smith and Eppinger, 1997a). For example, a design decision that is made downstream in the process may require a change in a decision that was made upstream in the process, which may be costly and time-consuming. This type of process is iterative: several passes may be required before successful completion of the project. It is important to note that while product development is inherently iterative by nature, and therefore, some iterations are necessary, unplanned iterations should be minimized. Planned iterations are necessary in product development as they help to refine design as it evolves; they also allow for experimentation and innovation to take place (Eppinger, 2001). These decisions must therefore be made very carefully. Thus, it is necessary to find efficient methods to help project managers and designers organize design tasks.

1.3 Management Methods and Tools for Product Development Projects

Because of the pressure to develop products in record times, managers search for ways to organize and coordinate the projects effectively in order to reduce development time. The conventional tools that are most commonly used to manage projects are PERT (Program Evaluation and Review Technique), CPM (Critical Path Method), Gantt Charts, and Microsoft (MS) Project. PERT and CPM are very useful graphical diagrams that can be used to describe work flow or information flow, as shown in Figure 1-1. In these diagrams, circles and arcs are used to represent activities and the information transferred between design tasks, respectively. When they are used as methods to determine the lowest cost, or the shortest path, etc., they describe work flow effectively. But PERT and CPM are usually used to represent sequential or parallel activities. To describe a complex design project with many iterations between tasks, a substantially large chart would have to be created that would span across several pages. Among these pages, it is difficult to exactly and clearly find feedback information from a downstream task to an upstream task.

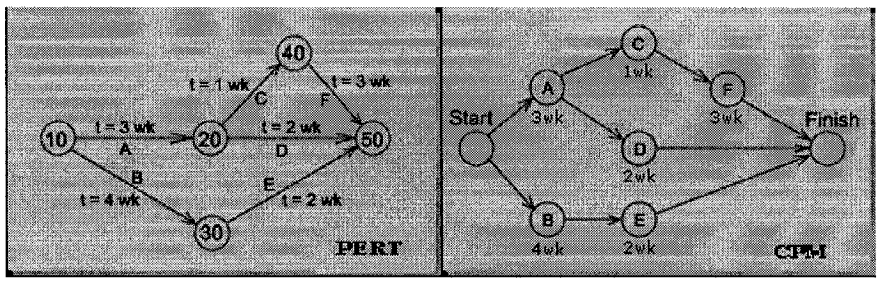


Figure 1- 1 Pert and CPM Network Diagram (adapted from http://www.netmba.com)

Similarly, while MS Project and Gantt Charts are powerful scheduling tools, they also cannot illustrate iterations within a process. Thus, a new method is needed to represent complex projects in a simple and efficient way.

1.4 The Design Structure Matrix

The Design Structure Matrix (DSM) has been proven to be a meaningful tool to represent and analyze complex design processes (Steward, 1981; Smith and Eppinger, 1997b). DSM is not based on work flows, but rather on information flows. As such, it does not require circles and arcs to link tasks and guide work flow. DSM uses informational relations between tasks and maps them into matrix form. It is a square matrix in which the columns and rows are identical since they represent the list of tasks needed to complete the design. The information relations are indicated in each cell in the following manner: a cell is marked if a task in a column provides information for an activity in a row, thus signifying dependencies between tasks. If a task in a row depends on information from another task, then the cell corresponding to that column is filled in. In this way, this matrix is an information pool where dependent and interdependent relations among tasks are evident. Therefore, while DSM includes functions similar to those of PERT and CPM, it also has the ability to describe feedback information. As a management tool, DSM organizes information flows among design tasks; using algorithms to manipulate the matrix, the methodology then determines an efficient sequence of tasks.

There are four types of DSMs that exist to represent data and solve problems in managing complex projects. Component-based DSM and team-based DSM are both static. Component-based DSM models components of a product such as parts of an

assembly or parameters of a part, and team-based DSM models groups and their interactions in an organization. The static DSMs are used to create an assignment plan for a project. Task-based DSM and parameter-based DSM are time-based, and they are the most popular applications of DSM. Task-based DSM is used to model processes based on information flows between tasks, while parameter-based DSM models relationships between design decisions and parameters.

The research and application presented in this thesis is based on a static DSM, in other words, the relations among design tasks remain unchanged with time in the analysis process. While this may not reflect what occurs in practice, the methodology has some powerful functions and provides a very good approximation to the sequencing of design tasks.

1.5 Objectives of the Thesis

The objectives of the thesis are to apply the DSM in the design process of an actual product. In doing so, a detailed process will be used to describe the application of DSM in the product development process, and the design process of the motorized wheel will be updated. The methodology will also explain how to quantitatively evaluate the resulting process. In the practical application, the DSM is used to organize the information of the actual design process in order to generate the execution sequence of design tasks. Its quantitative function will help to evaluate the development time.

The product to which the DSM method will be applied is a motorized wheel. In this product, an electric motor and transmission are in a highly compact form inside the housings of the wheel. The 24-V motor with a pure earth magnet generates torque and speed, and then power is transferred by the transmission into the body of the wheel. This

motorized wheel can be applied to many small means of transportation, for example, in an electric wheelchair. In the design of this product, the design process of the motorized wheel must be repeatable, and a brake system for the wheel must be added. In order to complete these, three DSM models of the design of the motorized wheel will be created:

- Model 1 is a parameter-based DSM used to analyze the dimensional relations of a single part.
- Model 2 is a task-based DSM used to define the design sequence of tasks for the motorized wheel.
- Model 3 is a task-based DSM built to add a new brake system.

During the design of a product, questions that often arise are how to organize available information, where to begin the design work, and what work needs to be done at every step. These questions will be answered in this thesis. The DSM will also be used to estimate the total design time for the motorized wheel.

The thesis is organized as follows. Chapter 2 presents a review of the literature in the field of product development. Chapter 3 discusses the DSM methodology. In Chapter 4, three design processes are used in the application of the DSM to the product development process. Chapter 5 introduces two calculation methods to coupled design tasks. In Chapter 6, the DSM methodology and its application will be concluded.

CHAPTER 2 LITERATURE REVIEW

In this chapter, a review of different approaches and their main contributions in the area of product development are presented. The DSM as a new methodology and its application in product development is introduced. The contribution of this thesis is then discussed.

2.1 Related Research

Researchers have studied various methods of integrated product development, including concurrent engineering, as a means of reducing development time for complex design projects. To achieve concurrency, preliminary information must be released to downstream tasks to allow for overlap of activities. However, due to uncertainty in the early stages of a product development process, the release of incomplete information to downstream tasks may potentially introduce the need for rework, resulting in unplanned task iterations, if a change occurs in upstream information. Thus, potential risks must be carefully examined to ensure that added time and effort are kept to a minimum (Krishnan *et al.*, 1997). Many researchers have studied the overlapping problem mostly using micro-models of the development process, consisting of an upstream activity and a downstream activity, and they describe the information exchange patterns between the two activities, as shown in Figure 2-1.

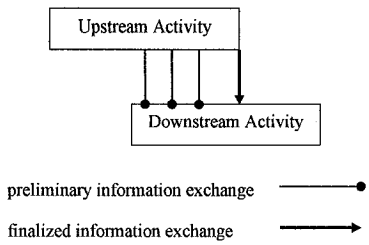


Figure 2- 1 Overlap Process

Krishnan *et al.* (1997a) have studied overlapping by developing a quantitative model in which iterations are modeled, and which is based on the speed of the evolution of information and the sensitivity of downstream activities to changes in upstream information. In their study, they identified three kinds of task execution patterns that occur in product development: sequential, parallel, and coupled. Each of these occur based on dependence, independence, and interdependence of information, respectively. Among these, the coupled tasks are most difficult to execute because of their interrelated relationships. The authors study the effect of overlapping tasks that have dependent, independent, and interdependent relations of information. They predict the optimal strategy for the amount of overlapping of activities.

Ha and Porteus (1995) have also studied an overlapped process. According to their research, communication is required frequently among team members in order to transform information frequently. A sufficient amount of information exchange can minimize time and improve quality. Loch and Terwiesch (1998) used a concurrent engineering model to study the overlapping of two coupled tasks. They study the trade-offs between the two tasks. They suggested that information sent at the right time to a downstream activity could reduce the rework of that activity caused by the future changes

of the upstream activity. Yassine *et al.* (1999) used the information schedule to explain the overlapping tasks. Partial overlapping exists in the interdependent tasks, and the information exchange between them has to minimize the risk of rework of downstream activity when engineering change occurs for the upstream activity.

Another set of researchers study product development as a system of interrelated elements, and use various methods to decompose integrated tasks and organize them by modeling the system in matrix form. While the representation of a system in matrix form is not new (Warfield, 1973), Steward (1981) developed the DSM based on network analysis techniques used in other fields and became the first to apply the method in product development processes. Originally, the matrix was used to show the relationships between the technical parameters of a design. The DSM became a management tool used to model and analyze relationships between components, teams, and activities. The latter type allows for the study of the three kinds of activities mentioned above, and can be used to assemble all the relative tasks of a process together and indicate their relations. The function of rearranging tasks with the DSM can help to distinguish the relations among tasks: dependent, independent, and interdependent. It then becomes apparent as to which tasks are sequential, parallel, or coupled.

Since the inception of this method, numerous authors have developed models based on the DSM. For example, iterations in the design process are an important consideration in these models. A number of models representing iterations have been developed using various methods based on the DSM (Smith and Eppinger, 1997; Ahmadi and Wang, 1994). The numerical evaluation of the DSM has been studied, making it possible to mathematically calculate and evaluate the design process in terms of its

expected duration for coupled tasks (Smith and Eppinger, 1997). Whitney *et al.* (1999) have used the DSM in order to capture knowledge to be used in an expert system. The DSM has also been simulated by Browning and Eppinger (1998) to minimize the probability of iterations or the project duration.

Many researchers have also contributed to the application of DSM since 1990s. The practical applications of DSM can be divided into four styles (refer to Section 1.4) according to different data that can be represented in a DSM. Pimmler and Eppinger (1994) constructed a component-based DSM in the design of an automotive climate control system. They built the relations of the components of the automobile climate system and found the blocks (formed by coupled tasks) to divide relative design tasks into groups. McCord and Eppinger (1993) applied a team-based DSM in an automobile engine design process. They analyzed the organizational structure necessary for an improved automobile engine development process. Their organizational analysis involved the level, frequency, direction, and timing of communication among design groups. Black *et al.* (1990) built a parameter-based DSM in an automobile brake system. This example revealed that DSM can be applied in the organization of the detailed work including space, material, energy, etc. Eppinger (2001) used task-based DSM in the analysis of semiconductor development at Intel. In this example, by studying the three types of task interactions in the development process, he revealed where efforts should be concentrated and where unplanned iterations that should be avoided.

2.2 Contribution of the Thesis

Although practical examples of DSM application in the product development process abound in the literature, few studies explain the details of application of DSM in the

design process. Therefore, in this thesis, in addition to the application of the DSM to an actual product, the theory and application rules are introduced to the reader. In this thesis, the tool is used to organize and guide the design process of a motorized wheel.

CHAPTER 3 DSM DETAILS

In this chapter, the details of the DSM are introduced. First, information dependencies that form the basis of the DSM methodology are discussed. Second, the details of the DSM methodology are described. Finally, the numerical DSM methodology is presented.

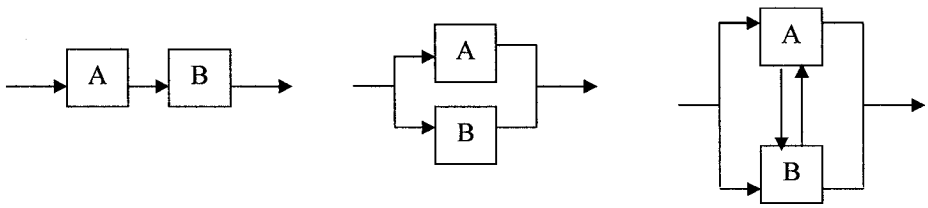
3.1 Information Dependencies

The product development process can be executed in a sequential manner, in parallel, or concurrently, i.e., in a coupled manner, or through a combination of these approaches. While some design tasks must be carried out sequentially or in parallel, other design tasks should be executed concurrently. In a complex process, the above three kinds of execution manners all exist at the same time. Which approach is best depends on the information relationships among product development tasks.

In the management of a product development process, the DSM method is based on information flows, rather than on work flow (Eppinger, 2001). Recall that a design process refers to a process of gathering and transforming information in order to find a solution to determine product parameters. For example, in the design of a gear, the surface hardness, interior hardness, and heat treatment technology are all types of information needed to make decisions on the type of material that is suitable for the gear; thus, the decision of the material is a result of processing the above information.

The information dependencies among tasks have a strong influence on the way a process can be managed. Three types of information relationships exist: dependent, independent, and interdependent. Dependent information refers a relationship where a downstream task can only begin when it receives information from the completed

upstream task, so the tasks with a dependent relationship are carried out sequentially. This is also referred to as sequential dependence, as shown in Figure 3-1(a), where, for an upstream task A and a downstream task B, information flows from task A to task B, but not vice versa. Independence of information refers to the relationship where a task can be executed without information needs from another tasks, so they are independent can be done simultaneously. Independence is shown in Figure 3-1(b), where no information exchange exists between A and B. Interdependent information, also called coupled tasks, refers to the relationship where tasks depend on each other for information. Interdependence is shown in Figure 3-1(c), where information flows from A to B and from B to A at the same time.



(a) Sequential Dependence (b) Independence (c) Interdependence

Figure 3- 1 Information Dependencies

The sequential or parallel design tasks are easily managed because the information flows among them are dependent or independent. These processes can be easily calculated mathematically. For example, in order to predict the design time for sequential tasks and parallel tasks, as shown in Figure 3-1(a) and 3-1(b), this can be calculated as follows:

Design time $= t_A + t_B$ (Sequential tasks) (3.1)

Design time = Max $(t_A + t_B)$ (Parallel tasks) (3.2)

Most design processes, however, involve interdependencies. If the coupled tasks are simply executed in sequential or parallel manners, they will lose necessary information exchange in the execution process, which will lead to a risk of quality loss. At the same time, the relationships among coupled tasks cause rework, or iterations among them. These planned iterations and information feedback between tasks are necessary to ensure that the design results in good quality. However, some rework, or unplanned iterations, is harmful because it takes more design time and cost. Thus, when relationships between tasks are decided upon, the true relations must be determined in order to form necessary or planned iterations and to avoid unnecessary or unplanned iterations which are the result of incorrect or unimportant relations. It is challenging to look for suitable and useful management tools to organize the design process to properly distinguish the relationships of design tasks.

3.2 Product Design Process Based on Information Flows

The DSM is different from conventional tools in that it requires the information needs among tasks in order to determine the best ordering of tasks.

3.2.1 Constructing a DSM

In a DSM, design tasks are placed sequentially in the rows and columns, and the information requirements among tasks are expressed using a 0 (or an X, or any other arbitrary notation). An information flow can be expressed as a_{ij} in the matrix, designated as A, which means that the task on the i_{th} row requires information from the task on the j_{th} column. a_{ij} does not have a quantitative value, but rather, it means that for task i, an

information input is demanded from task j. Therefore, the three kinds of information flows shown in Figure 3-1 can be expressed in a DSM as shown in Figure 3-2. The sequential relationship between task A and task B is described as Figure 3-2(a). Task B is carried out sequentially after task has been finished. Task B requires an input from task A, so a 0 is put in the blank a_{21}. In Figure 3-2(b), task A and task B are parallel, so the blanks are empty to express that they do not have information exchange. The interdependent relationship between the two tasks is shown as Figure 3-2(c). They exchange information to each other. Therefore, the 0s are placed in the blanks a_{12} and a_{21} to indicate that they receive information from each.

	A	B
A	*	
B	0	*

(a) Sequential Dependence

	A	B
A	*	
B		*

(b) Independence

	A	B
A	*	0
B	0	*

(c) Interdependence

Figure 3- 2 Equivalent Matrix of Information Flows in a DSM

The DSM is a good method to describe and solve the iteration process of a project. Obviously, for a complex design process including iterations, it is very difficult for certain methods to describe them. For example, it is impossible to use a decision tree

to represent the interrelationships in design tasks, and CPM or PERT is effective for purely sequential or parallel ordering tasks, but not for coupled tasks. However, DSM can easily solve this kind of problem.

Product design is a complex system which needs to be modeled in order to understand it well. The key philosophy of DSM is to assemble all design information into an easily understandable model. The typical process of modeling a design project can be undertaken as follows:

(1) Divide the project into individual sub-tasks;

(2) Express the relationships of sub-tasks;

(3) Note the importance of each relationship (input or output).

In this way, a project can be clearly expressed by DSM through rows that indicate all inputs that are required for the tasks in that row, and columns that denote all outputs that are given by the tasks in that column.

In order to illustrate how to build a DSM, an example of the product development process of a mug cap is used (Baldwin and Clark, 2000). Before drawing the DSM, there are two prerequisites. The first prerequisite is to identify all the tasks that take place in this design process. In this design, only four tasks are involved, namely the determination of the Cap Shape, Cap Diameter, Vessel Diameter, and Cap Material. Although the Vessel Diameter does not belong to the mug cap, the decision of Vessel Diameter and Cap Diameter are interrelated. The other three tasks are the basic parameters required to design the mug cap. The second prerequisite is to determine the information flows among tasks. This step is time-consuming, and must be carefully done in order to avoid mistakes. The reason is that a big design project normally is executed by often hundreds of people.

A single person cannot understand every detail involved in a complex project. Therefore, information gathering should be carried out by consulting all individual members and teams involved in the project.

Once all of the information has been obtained, the DSM can be built. In the case of the mug cap, first, the four tasks in the design process are listed. Next, a square matrix is formed by arranging the four tasks in rows and columns. Each row corresponds to a task, and the 0s that will be marked in this row refer to the information that this task receives. Empty blanks mean that this task does not get information input from those tasks in the corresponding columns. In the same manner, each column also refers to a task, and the 0s in the column represent the output that this task gives to the other tasks. Again, empty blanks in a column denote that this task does not give out information to those corresponding tasks in rows. Finally, relationships between tasks are represented by 0s in the matrix. The diagonal blanks are marked by using asterisks since a task cannot give information to itself. The completed DSM is shown in Figure 3-3.

	1	2	3	4
1 Cap Shape	*	0		
2 Cap Diameter		*	0	
3 Vessel Diameter		0	*	
4 Cap Material				*

Figure 3- 3 Design Structure Matrix of a Mug Cap

3.2.2 Reading the DSM

Before analyzing the DSM, the order of the tasks must be rearranged by moving 0s on the right of the diagonal as much as possible to the left of the diagonal because 0s in the

upper diagonal are the reason that rework occurs. A new DSM emerges as shown in Figure 3-4. This rearrangement does not change any information between tasks, but simply modifies the order of the tasks listed.

	2	3	4	1
2 Cap Diameter	*	0		
3 Vessel Diameter	0	*		
4 Cap Material			*	
1 Cap Shape	0			*

Figure 3- 4 The Rearrangement of the Matrix

From the DSM in Figure 3-4, it can be seen that the matrix is divided by asterisks into two sections: an upper diagonal and a lower diagonal. The 0s in the lower diagonal represent feed-forward information exchanges in which the earlier tasks give inputs to the later tasks and they can be executed sequentially. For example, task 1 needs an input from task 2, so it has to be executed after task 2. In contrast, the 0s in the upper diagonal denote feedback information exchange in which the later tasks may generate rework to the earlier tasks and they form coupled tasks. For instance, since task 3 requires an input from task 2 in the lower triangle, task 3 should be executed after task 2, but task 3 also gives an output to task 2 in the upper diagonal that leads to a rework of task 2. Therefore, task 2 and task 3 are coupled and they have to be executed concurrently.

3.2.3 Partitioning and Tearing the DSM

Partitioning is the process of reordering rows and columns of DSM in order to move feedback 0s in the upper diagonal area to the lower area or as close as possible to the

diagonal. The goal of partitioning is to reduce feedback as much as possible. The expected result using the partitioning method is to convert the DSM into a form such that all 0s are in the lower diagonal area, or that they take shape of several blocks along the diagonal (as in Figure 3-4).

The unpartitioned DSM is the original matrix that gathers information on the design process. Normally, it is not convenient to make conclusions from this matrix because feedback information interferes with the analysis. From the comparison between Figure 3-3 and Figure 3-4, the 0 in a_{12} in Figure 3-3 is moved into the lower diagonal by reordering the task sequence. Thus, task 1 can be carried out sequentially. As a result, partitioning efficiently removes unnecessary rework information and it benefits to analyze the correct execution sequence. But then a new question is raised on how to execute tasks within iterations.

Tearing is the process of looking for feedback information and regarding this information as being known. Within the coupled tasks in Figure 3-4, the forward information (a_{32}) demands a sequential execution between task 2 and task 3. But the backward information (a_{23}) requires rework to task 2. If a_{23} were assumed to be known, clearly, the tasks in the block could be carried out in a sequential order. The backward information, chosen as being known, is called a tear. For a complex block formed by coupled tasks, one or more tears are required to be chosen to break the iterations among tasks. After tearing, the newly partitioned DSM will be more like a lower triangular matrix, shown as Figure 3-5, where the 5 is chosen as an arbitrary notation to indicate that this information is now assumed to be known. Because tears are the assumptions that designers use, a minimal number of tears is recommended to tearing process. Normally,

the longest information circuit method is used in the tearing process by the purpose to make a minimal assumptions (refer to Appendix).

	2	3	4	1
2 Cap Diameter	*	5		
3 Vessel Diameter	0	*		
4 Cap Material			*	
1 Cap Shape	0			*

Figure 3- 5 The coupled Tasks are torn (5, a known information mark)

3.2.4 Modelling the Design Process with Iterations

Traditionally, the design process was carried out as a sequential process. This ignored the interdependencies between tasks. But iterations between two tasks as shown in Figure 3-1(c) are often necessary in the design phase. For those tasks that are strongly interrelated, executing them as sequential tasks would result in a quality loss (Krishnan, *et al.*, 1997b). Because for those tasks that are coupled, they should be carried out concurrently in order to feed information to each other in every step until they have converged. It takes more time to execute coupled tasks sequentially because overlap execution saves time by doing them concurrently (Krishnan, *et al.*, 1997).

The DSM model is a representation and analysis tool to identify the ordering of tasks and to recognize the main effects on design time and quality. A partitioning algorithm of pushing marks to the left bottom triangle is used in a DSM model to reorder the sub-tasks in the matrix. As a result, the sequence of executing sub-tasks is generated, i.e., which tasks are carried out in sequential order, in parallel order, or in concurrent order. A DSM model is also analytically tractable to help to observe the relationship

between the structure of the problem and the development time of the project. An example of a camera design is used to illustrate the DSM functions (Smith, 1993). Shown in Figure 3-6, it is a task-based DSM that consists of eight design tasks for the development process of a camera. From the matrix, the design process is divided into three sections. Task A and task B belong to the first section. Task A sets up the basic characters of the camera, such as functions, usage, etc. Task B transfers theses specifications from task A to design concepts. Task B should be carried out after task A. After execution of section 1, section 2 and 3 can be simultaneously carried out. Section 2 focuses on the design of mechanical structure of the camera, and section 3 refers to the design of the lens mechanism of the camera. In section 3, task G and task H should be completed sequentially because the lens housing should be designed after the lens has been decided upon. In section 2, four design tasks-C, D, E, and F are coupled to form a block. They are correlated in space and function, so they need feedback information from each other. Thus, the four tasks have to be finished concurrently.

Based on the above, the total design time T can be estimated by:

$$T = a + b + \max \{f(c, d, e, f), g + h\} \tag{3.3}$$

where a, b, c, d, e, f, g, and h represent the execution time when each task is finished alone. The function f(c, d, e, f) describes to the execution time of the block.

		A	B	C	D	E	F	G	H
Section 1	A (Set Specifications)	*							
	B (Design Concept)	0	*						
Section 2	C (Design Shutter Mechanism)		0	*					
	D (Design Viewfinder)		0		*				
	E (Design Camera Body)		0			*			
	F (Design Film Mechanism)		0				*		
Section 3	G (Design Lens Optics)		0					*	
	H (Design Lens Housing)		0					0	*

Figure 3- 6 Camera Design (Smith, 1993)

3.2.5 Improving Information Flow Using DSM

The design process using DSM is a process of manipulating information that is known. Therefore, the improvement of the design process is also a process of improving information flows. In this section, the advantages of applying DSM to information flows are discussed.

- Rearranging Tasks to Reduce Rework

In Section 3.2.3, tasks were rearranged in order to present the information in the DSM more clearly. The goal of the rearrangement is to reduce the number of feedback loops, namely 0s in the upper diagonal. Comparing Figure 3-3 and Figure 3-4, it can seen that one 0 in the upper diagonal is eliminated, which means that potentially unnecessary rework is avoided.

Pushing 0s as much as possible to the lower diagonal is achieved by rearranging the rows that receive more inputs to the bottom of DSM and rearranging columns that give more outputs to the left side of DSM. In this way, the feedback information is avoided as much as possible. These 0s, which cannot be moved to the left triangle, form

one or more blocks along diagonal. Thus, based on the information in the matrix, all design tasks can be divided into three groups: sequential tasks, parallel tasks, and coupled tasks. The example in Figure 3-3 is easily reordered, since, for a small DSM only with a few tasks, it is easy to reorder the sequence of tasks according to DSM rules (refer to Section 3.2.2). However, the DSM is normally used to arrange large product development projects, so it is less straightforward to rearrange so many tasks and their information exchanges without the help of a computer. PSM32 is a user-friendly software designed to help in partitioning (reordering), tearing, and generating suggested execution sequences. This software will be widely used in the design processes in Chapter 4. Details of the software can be found in Appendix A.

- Reorganize Tasks to Avoid Unplanned Iterations

To illustrate this point, consider the information flow of a badly organized ten-task project presented in Figure 3-7.

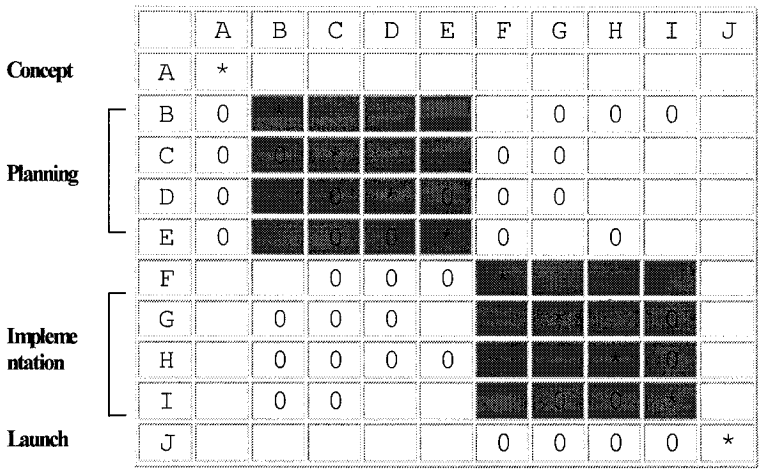


Figure 3- 7 A badly-organized DSM reveals unplanned iterations (Eppinger, 2001)

After reordering, the sequence of tasks remains unchanged. This design is divided into four phases: concept, planning, implementation, and launch. Along the diagonal of the matrix, blocks of coupled tasks are drawn according to design phases. Only partial 0s found in the upper diagonal are included in two blocks, and the others are called unplanned iterations, which are undesirable during design as they cause wastage of design resources. Thus the matrix should be reorganized.

(3) Reduce Iterations

The content of some tasks can be changed to break down coupled tasks, or decouple them, into smaller sets (blocks or tasks). In this way, the number of tasks and people are increased, but the number of unplanned iterations is efficiently reduced. In order to decouple iterations, the main knowledge between teams can be transferred. This method can be achieved by adding to each design team someone who knows the tasks of other teams, or using simulation software to forecast the need for information. For example, during the design of a can opener, CAD software can be used to assemble the parts to form a complete product and then the functionality can be simulated. In this way, it can be determined if each part is suitable for the whole completed product and if interference exists among these parts. Using computer software, some information can be obtained early, so unplanned iterations can be removed. Secondly, a new task can be added earlier to introduce something common in advance to the coupled tasks. For instance, in the design of the mug cap, the Cap Diameter and Vessel Diameter are coupled tasks. If a task of Recommended Diameter is added, the coupled tasks receive information from it, and they can be broken and be carried out in parallel. It is possible to break this kind of iterations by looking for recommended dimensions in professional handbooks. Finally,

tasks can be redefined within blocks. It is similar to the reorganization discussed in point (2) above. Through adding one or two tasks, unplanned iterations can be eliminated. While iterations are needed to improve the quality of design, decoupling iterations can help to increase the speed of design.

3.3 Four Types of DSMs

Four different types of DSM exist to represent data and solve problems in engineering management. They are component-based DSM, team-based DSM, task-based DSM, and parameter-based DSM, as shown in Figure 3-8.

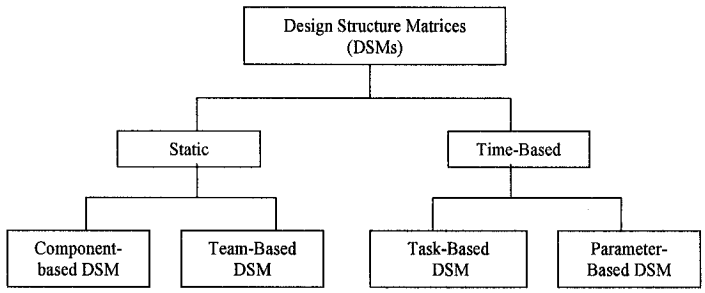


Figure 3- 8 DSM Taxonomy (Dorf, *et al.*, 1999)

Component-based DSM and team-based DSM are both static. Component-based DSM represents components of a product such as parts of an assembly or parameters of a part, and team-based DSM represents groups in an organization. An example is given by Pimmler and Eppinger (1994) using a clustering algorithm to analyze a component-based DSM as shown in Figure 3-9, which shows a design process for an automotive climate control system that consists of 16 participants, and the iterations between them are shown in the DSM.

		A	B	C	D	E	F	G	H	I	J	K	L	N	M	O	P
Radiator	A	*	0														
Engine fan	B	0	*														
Heater Core	C			*													0
Heater Hoses	D				*												
Condenser	E		0			*	0		0								
Compressor	F					0	*		0	0							
Evaporator Case	G							*									0
Evaporator Core	H					0	0		*	0							0
Accumulator	I						0		0	*							
Refrigeration Controls	J										*						
Air Controls	K											*					
Sensors	L												*				
Command Distribution	M													*			
Actuators	N														*		
Blower Controls	O															*	0
Blower Motor	P			0				0	0							0	*

Figure 3- 9 A component-based DSM for an Automotive Climate Control System (Pimmler, *et al.,* 1994)

Reordering the DSM, the new result is shown in Figure 3-10, which provides a cluster of blocks, called trunks that are formed by coupled participants. Based on this, an optimal team assignment can be divided to three teams as shown as Figure 3-10 by the colored blocks. Of course, if this design were regarded as a small one for one designer, the three blocks could be thought of as three groups deciding on the design parameters.

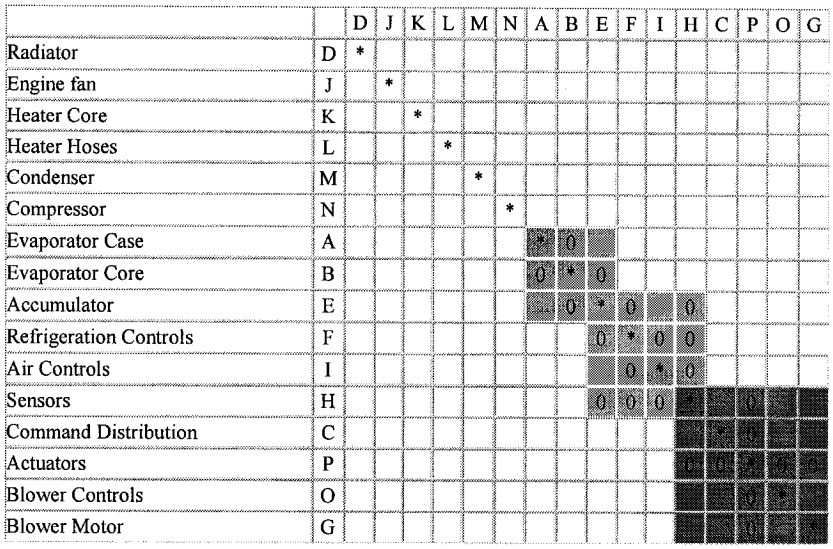


Figure 3- 10 A Cluster of Blocks Appear along the Diagonal after Reordering

Task-based DSM and parameter-based DSM are time-based, and they are the most popular applications of DSM. Task-based DSM is especially widely used to determine the optimal design order among many sub-tasks in order to master the most efficient design mode, where a design mode refers to a closely-related group of tasks. The Camera Design in Figure 3-6 is a good example of task-based DSM. From the DSM, it is very clear which tasks are executed in sequence, parallel, or coupled.

Parameter-based DSM uses parameter interrelationships to analyze system architecture. For example, when a gear (in a transmission) is to be designed, the module, press angle, helix angle, and so on, are needed to calculate the dimensions of the gear. These parameters exchange data using mathematical algorithms. Moreover, when gears, shafts, and housings are put together to form a transmission, each part, for example a gear, is a component of the assembly of the transmission, so those parts have spatial relationships in the design process. Partial parameters for Gear Design are assembled in

Figure 3-11 to form a parameter-based DSM. The module, teeth number, press angle, and helix angle are the basic given parameters, and the head height coefficient and foot head coefficient can also be found in a handbook. Then, the tasks are to calculate those parameters such as the cross diameter and so on. This example shows information exchange between parameters.

	1	2	3	4	5	6	7	8	9
1 Module	*								
2 Teeth No.		*							
3 Press Angle			*						
4 Helix Angle				*					
5 Head Height Coe.					*				
6 Foot Height Coe.						*			
7 Cross Diameter	0	0		0			*		
8 Head Diameter	0				0		0	*	
9 Foot Diameter	0					0	0		*

Figure 3- 11 Partial Parameters in Gear Design

3.4 Numerical DSM

As discussed, one application of the DSM that is different from conventional management methods is its numerical function. Normally, when a design process is evaluated, the focus is on the lead time of the design process. Referring to Formulas 3.1 and 3.2, parallel and sequential tasks are easily given a quantitative calculation, but the iterative mode requires complex mathematical algorithms.

3.4.1 Work Transformation Matrix

In order to build a numerical DSM model, a work transformation matrix (WTM) is introduced by Smith (1993) and Eppinger (1997) to help develop the design iteration model. The WTM is an extended numerical version of a fully coupled design structure (Smith and Eppinger 1997).

The information in a WTM as shown in Figure 3-12 is defined as follows:

- The off-diagonal elements represent the strength of dependence between tasks;
- The diagonal elements in the WTM represent the time that it takes to complete each task in isolation.
- Time in other sub-stages is a function of the amount of time spent in the previous stage.

	A	B
A	4	0.2
B	0.4	7

Figure 3- 12 Work Transformation Matrix

In order to have a linear analysis, three assumptions are made in the WTM model:

- All tasks are executed in parallel in several stages;
- Rework is a linear function of work in the previous iteration;
- Rework and activity durations do not vary with time.

The first assumption means that coupled tasks should be executed concurrently in stages. In each stage, every task is finished in parts. These stages are parallel, so that the total amount of work can be easily added together (refer to Formula 3.6 in next section).

The second and third assumptions are for the convenience of mathematical calculation. In this way, a stable linear equation can be created to represent the portions that tasks will be finished in each stage (refer to Formula 3.4 in next section).

3.4.2 Work Vector

A work vector u_t, used to describe the WTM model, is an n-vector, where n is the number of coupled tasks to be completed. Each element in this vector is the amount of work to be done on each task after iteration stage t. The initial work vector u_0 describes the beginning status of the iteration process, and it is a vector of 1's, which indicates that at the start of the iteration process, all the work for each task remains to be finished. In each iteration stage, all sub-tasks are completed in some proportion. For a sub-task, its execution in this stage causes some rework to other sub-tasks that have dependencies with it. The most work of a sub-task is done in the first, second, or third stage. In each iteration stage, the work vector can be calculated by this formulation:

$$u_{t+1} = A\, u_t \tag{3.4}$$

A is a matrix whose elements a_{ij} represents the amount of rework for sub-task i after doing one unit of sub-task j. The matrix A describes the strength of the dependencies portion of the WTM and its elements are those off-diagonal numbers in Figure 3-12. Thus, the work vector can be expressed as:

$$u_t = A^t\, u_0 \tag{3.5}$$

If the design process can be divided into M stages, there is a work vector in every stage. The sum of those work vectors is called the total work vector U.

$$U=\sum_{t=0}^{M} u_t = \sum_{t=0}^{M} A^t u_0 = (\sum_{t=0}^{M} A^t)\,\mathbf{u}_0 \tag{3.6}$$

From this formulation, the total amount of rework for a task can be found. For example, if task i has a number 1.52 in vector U, a total of 52% rework on task i will be required in subsequent stages.

3.4.3 Decomposition of Matrix A

Recall that it is assumed that rework is a linear function of work done in the previous iteration. The matrix A, built by elements that represent rework, can be decomposed into:

$$A = S \Lambda S^{-1} \tag{3.7}$$

where Λ is a diagonal matrix of the eigenvalues of A, and S is the corresponding eigenvector matrix (S is invertible). Therefore, the total work vector

$$U = (\sum_{t=0}^{M} A^t)\,\mathbf{u}_0 = S(\sum_{t=0}^{M} \Lambda^t)S^{-1}\,\mathbf{u}_0 \tag{3.8}$$

Normally, any eigenvalue of matrix A is expected to be less than one because if so, the design process converges and is stable. If the maximal eigenvalue is not more than one, the total work vector is bounded as the number of stages increases. Therefore,

$$\lim_{M\to\infty} \sum_{t=0}^{M} \Lambda^t = (I-\Lambda)^{-1} \tag{3.9}$$

$$U = S(I-\Lambda)^{-1} S^{-1} u_0 \tag{3.10}$$

3.4.4 A 3x3 DSM Example

A small DSM model of 3x3 is used to describe the application of the numerical DSM in the analysis of the design process. Three coupled sub-tasks are taken out from a DSM because they have strong interrelationships as shown in Figure 3-13. Values are given to

interpret the eigenstructure. As shown in matrix A, the off-diagonal values denote the probabilities of rework. For instance, a_{12} is 0.2, which means that doing one unit of task 1 causes 20% probability of rework to task 2.

	1	2	3
1	*	0	
2	0	*	0
3	0	0	*

Figure 3- 13 A coupled Block with 3 Tasks

$$A = \begin{bmatrix} 0 & 0.2 & 0 \\ 0.4 & 0 & 0.5 \\ 0.3 & 0.1 & 0 \end{bmatrix}$$

Even though a coupled design task is may be completed in infinite stages for iteration, most work is finished in the first three or four stages. According to Formula 3.5, the first four work vectors are calculated as follows:

$$u_0 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \quad u_1 = \begin{bmatrix} 0.2 \\ 0.9 \\ 0.4 \end{bmatrix} \quad u_2 = \begin{bmatrix} 0.18 \\ 0.28 \\ 0.15 \end{bmatrix} \quad u_3 = \begin{bmatrix} 0.056 \\ 0.147 \\ 0.082 \end{bmatrix}$$

These results give the time remaining to finish coupled tasks. Comparing the results of u_1, u_2, and u_3 shows that the order of tasks contributes to the total work as follows, from the most to the least: task 2, task 3, and then task 1. This conclusion will be verified after calculating the total work vector.

According to the formula $A = S\Lambda S^{-1}$, the eigenvalue matrix Λ and eigenvector matrix S of matrix A are:

$$S = \begin{bmatrix} 0.370 & 0.472+0.286i & 0.472-0.286i \\ 0.821 & -0.717 & -0.717 \\ 0.434 & -0.059-0.422i & -0.059+0.422i \end{bmatrix}$$

$$\Lambda = \begin{bmatrix} 0.444 & 0 & 0 \\ 0 & -0.222+0.134i & 0 \\ 0 & 0 & -0.222-0.134i \end{bmatrix}$$

The three eigenvectors correspond to the three columns in the matrix S, and they are associated with the eigenvalues along the diagonal of the matrix Λ. The total work vector is:

$$U = S(I-\wedge)^{-1}S^{-1}u_0 = \begin{bmatrix} 1.488 \\ 2.441 \\ 1.691 \end{bmatrix}$$

Inspecting the total work vector shows that task 2 contributes most to the total work, then task 3, and finally task 1.

3.4.5 Conclusion

In this section, the numerical method based on the contribution to the convergence of the design process is discussed. The work vector and total work vector are calculated with

the strength of rework probabilities, and their results reveal the speed in which the coupled design tasks are finished.

The example in Section 3.4.4 shows a calculation process of three coupled tasks. According to the results, the three tasks can be ranked according to the design time consumed.

CHAPTER 4 DESIGN PROCESS OF A MOTORIZED WHEEL

In this chapter, the application of the DSM in the design of a motorized wheel is discussed. Three DSM models are created, one for part design, one for assembly design, and the last one for the design of a brake system.

4.1 Model 1 – Using DSM in Dimensional Decisions

In a graphical design process, dimensions are the main aspect that should be considered. The design of the motorized wheel requires about 45 parts. Here, rather than introducing the design process of each part, a key part, the counter gear 2 as shown in Figure 4-1, is taken as an example to discuss the application of the DSM in the decision of dimensions. The counter gear 2 is a part in the transmission group of the motorized wheel. The power (speed and torque) generated in the motor is delivered by the driven gear. The speed and torque are transferred by the counter gear 1 and counter gear 2 to the driven gear; in the end, they are passed to the output flank. As shown in Figure 4-2, the counter gear 2 works with the counter gear 1 and the driven gear to modify the ratio of speed and torque.

The DSM methodology is applied to generate a plan to guide the dimensional decisions of this part. Since this part includes so many dimensions that the DSM will be overly complex, only the axial dimensions of the counter gear 2, which are critical design dimensions, are used in this model.

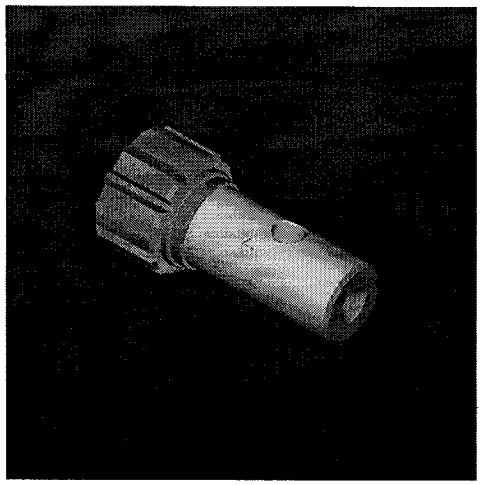


Figure 4- 1 Counter Gear 2

4.1.1 Constructing the DSM Model for Part Design

The assembly position of counter gear 2 is shown in Figure 4-2, while the axial dimensions of this part are shown in Figure 4-3. Since the axial dimensions of counter gear 2 are to be decided upon, the components which have relationships with it in the axial direction should also be considered, as shown in Figure 4-4.

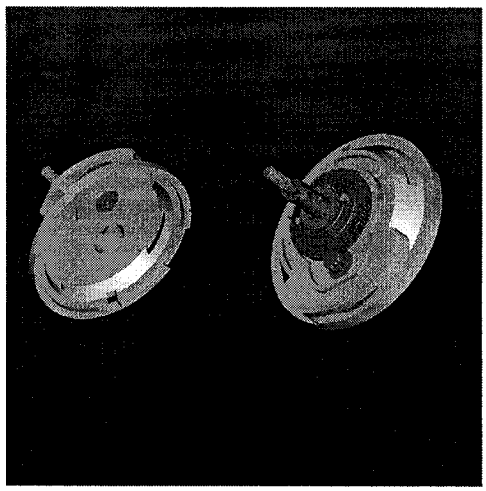


Figure 4- 2 Counter Gear 2 in the Complete Transmission

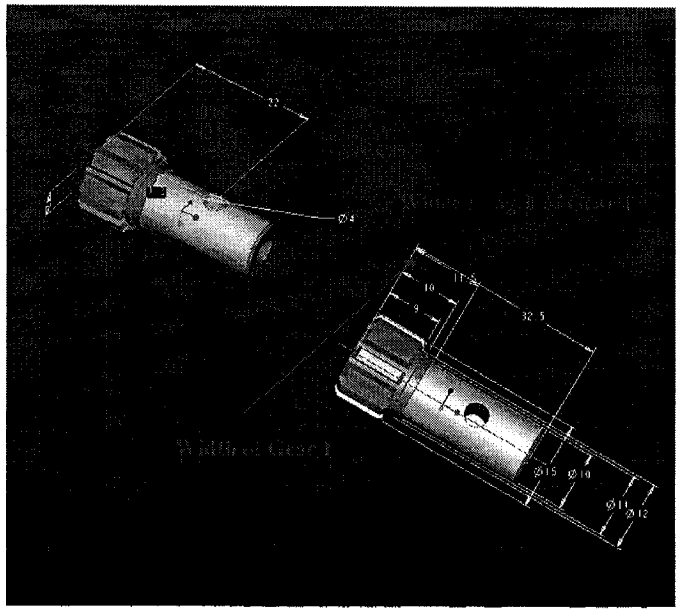


Figure 4- 3 Axial Dimensions of Counter Gear 2

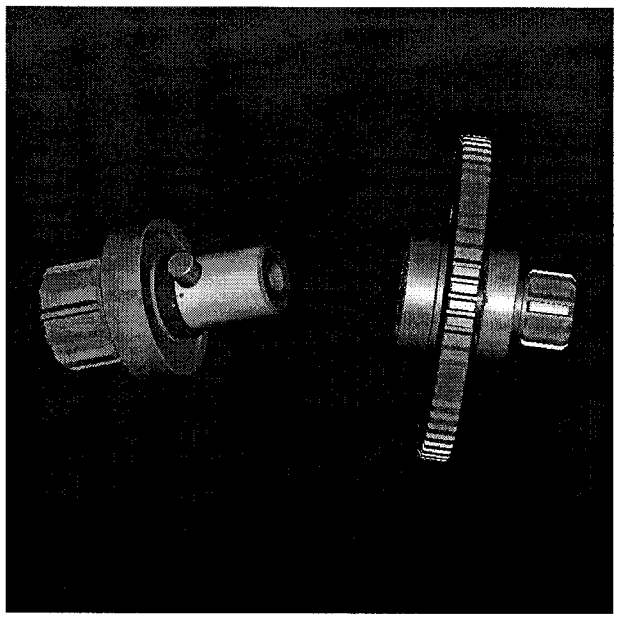


Figure 4-4 (a) The Components fixed with Counter Gear 2

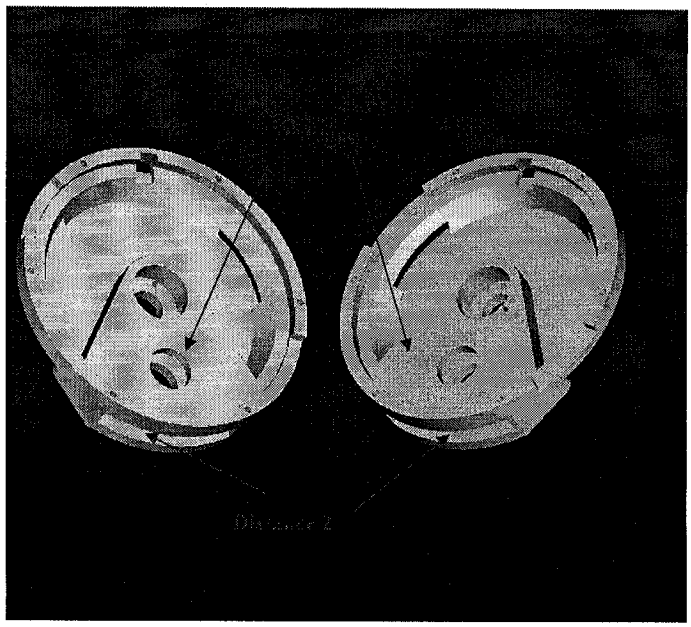


Figure 4-4 (b) The fixing Positions of Counter Gear 2 on the Transmission Housing

Figure 4- 4 The Parts that are relative to the Axial Dimensions of Counter Gear 2

Based on the above information, a parameter-based DSM will be built on the axial dimensions of counter gear 2. The first step is to list all of the related components that are important in deciding these dimensions. The list is shown in Table 1.

Table 4- 1 Relative Axial Dimensions

Parameter	**Symbol**
Whole Length of Counter Gear 2	x1
Width of Counter Gear 2	x2
Length of the Shaft	x3
Position of the Hole	x4
Diameter of the Hole	x5
Width of Washer Ring	x6
Width of the Hub of Counter Gear 1	x7
Distance 1 (on Transmission Housing)	x8
Distance 2 (on Transmission Housing)	x9

The next step is to construct a DSM as shown in Figure 4-5, whose rows and columns match the items in Table 4-1. In the matrix, all relationships between parameters should be carefully evaluated, analyzed and inserted into the proper entries. The usage of 0s represents the information flows. The 0s in a row indicate all the input information for deciding the parameter in that row, and the 0s in a column express the output information that the parameter in this column generates for deciding other parameters. For example, the 0 in a_{43} means that when parameter 4 (position of the hole) is to be decided, an input from parameter 3 (length of the shaft) is demanded.

	x1	x2	x3	x4	x5	x6	x7	x8	x9
x1 Whole Length of Counter Gear 2	*	0	0						
x2 Width of Counter Gear 2	0	*	0						
x3 length of the Shaft	0	0	*					0	0
x4 Position of the Hole			0	*			0		
x5 Diameter of the Hole					*		0		
x6 Width of Washer Ring			0			*	0		
x7 Width of the Hub of Counter Gear 1			0				*		0
x8 Distance 1 (on Transmission Housing)			0					*	0
x9 Distance 2 (on Transmission Housing)								0	*

Figure 4- 5 A 9*9 DSM for the Dimensions Decision

4.1.2 Partitioning and Tearing the DSM Model

Once the DSM has been initially built, it should go through the partitioning and tearing processes according to the methodology outlined in Chapter 3. In other words, the feedback information (0s in the upper area) should be moved to the left side of the diagonal by arranging rows and columns. The partitioning process and tearing process can be implemented with the help of the software PSM32 (refer to Appendix).

	x1	x2	x3	x8	x9	x7	x4	x5	x6
x1 Whole Length of Counter-Gear 2									
x2 Width of Counter-Gear 2									
x3 length of the Shaft				0	0				
x8 Distance 1 (on Transmission Housing)			0						
x9 Distance 2 (on Transmission Housing)									
x7 Width of the Hub of Counter-Gear 1			0		0	*			
x4 Position of the Hole			0			0	*		
x5 Diameter of the Hole						0		*	
x6 Width of Washer Ring			0			0			*

Figure 4- 6 The Partitioned Result of the DSM in Figure 4-5

After partitioning, as many 0s as possible are moved to the lower diagonal or close to the diagonal. Along the diagonal, two blocks are formed by coupled tasks, as shown in Figure 4-6. But two 0s in the upper diagonal area are not included in the two the blocks. They are unplanned iterations, so the structure of the DSM needs to be reorganized. The two 0s are outputs from parameter x8 and x9 to x3, so the best way is to go back to the original matrix in Figure 4-5 to check the relations among them. First, in this design process, x3 has the same length as x9, and x8 has a mathematical relationship with x9. Therefore, if x3 is known, the length of x8 and x9 are also automatically known.

Second, since x9 has the above relationship with x3, the output from x9 to x7 can also be replaced by the output from x3 to x7. In conclusion, the two parameters, x8 and x9, can be cancelled from Figure 4-5. The cancellation causes a new DSM to be formed, as shown in Figure 4-7(a). After partitioning, most 0s are moved into the lower diagonal, and the rest of the 0s form an iteration block along the diagonal, as shown in Figure 4-7 (b).

	x1	x2	x3	x4	x5	x6	x7
x1	*	0	0				
x2	0	*	0				
x3	0	0	*				
x4			0	*			0
x5					*		0
x6			0			*	0
x7			0				*

(a) Original Matrix

	x1	x2	x3	x7	x4	x5	x6
x1	*						
x2		*					
x3			*				
x7			0	*			
x4			0	0	*		
x5				0		*	
x6			0	0			*

(b) Partitioned DSM

Figure 4- 7 The New DSM for the Dimension Decision of Counter Gear 2

Parameters x4, x5, x6, and x7 can be executed in sequence or in parallel. But the execution of the three parameters in the block is unknown. The three items in this block should be carried out concurrently, but where shall designers start to execute iterations among them? Thus, this block has to be torn at first. It is easy to choose tears for the block because the three parameters contribute the same efforts to the whole design. The parameters x1 and x2 are successfully chosen to be torn points, as shown in Figure 4-8. The digits "2" and "1" are the tears – the feedback information that is supposed to be known.

	x3	x2	x1	x7	x4	x5	x6
x3	*	1	2				
x2	0	*	2				
x1	0	0	*				
x7	0			*			
x4	0			0	*		
x5				0		*	
x6	0			0			*

Figure 4- 8 Tearing the DSM at x1 and x2

4.1.3 Execution of the Design Process

One of the advantages of the DSM is that it can reveal an efficient sequence of tasks for the design process. The sequence of decisions regarding axial dimensions of counter gear 2 can be described by a production flow chart as shown in Figure 4-9. First, the three dimensions of x1, x2, and x3 are decided concurrently. The dimension decisions among them can be generated after executing them by three or more iterations in sequence from x3, to x2, to x1. After several iterations, the dimensions will be the expected values. Second, when the dimensions in the block are decided upon, it will then be possible to sequentially determine the parameter. Finally, the last three parameters can be settled in parallel based on the inputs from the block and x7.

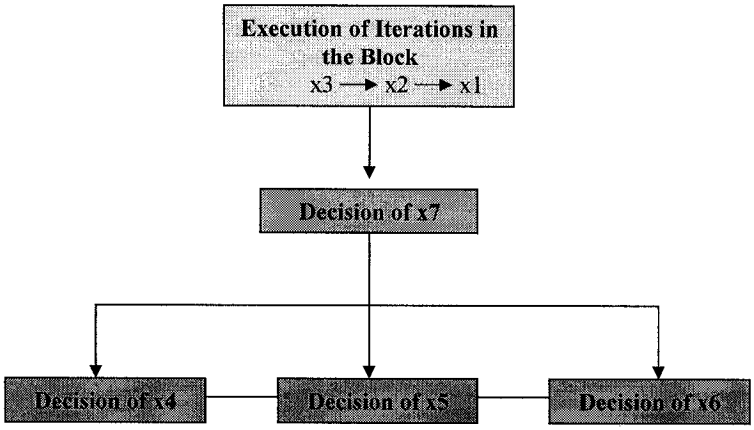


Figure 4- 9 Production Flow Chart of Implementation of Tasks

In this model, the DSM helps to draw up an efficient plan for a part design. From the result, it clearly tells designers where to start with the design and what should be considered for a parameter. Similarly, the same work can be done to all parts in the motorized wheel, including graphic design and parameter decisions such as deciding upon materials to be used.

4.2 Model 2 – Constructing the DSM Model for Assembly Design

Partial structures of the motorized wheel have been shown in last section. The whole view of the motorized wheel can be seen Figure 4-10. Both sides of the complete wheel are shown in Figure 4-10(a). There are two assemblies inside the wheel-transmission and motor, which can be seen in Figure 4-10(b) and 4-10(c).

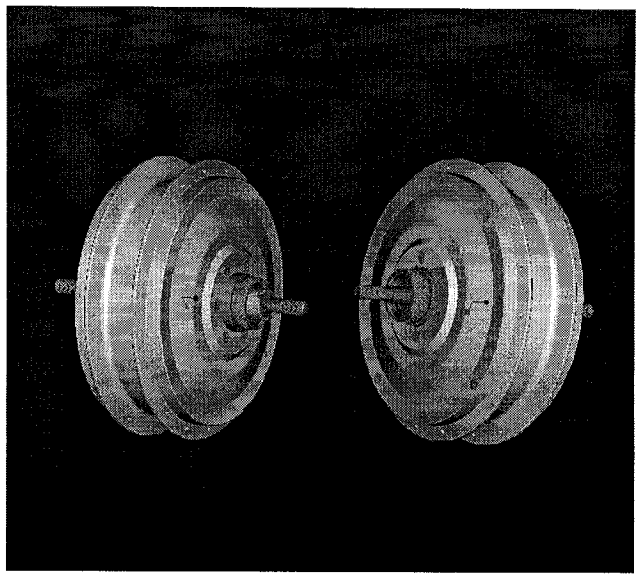


Figure 4-10(a) The Motorized Wheel

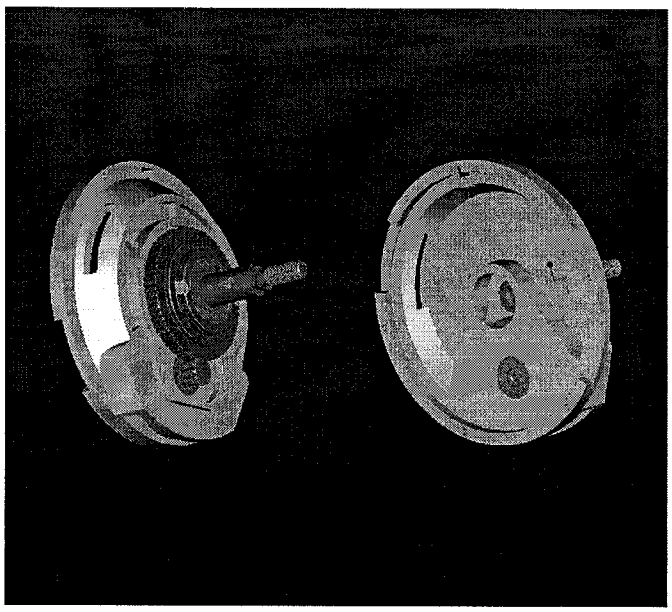


Figure 4-10(b) The Complete Transmission

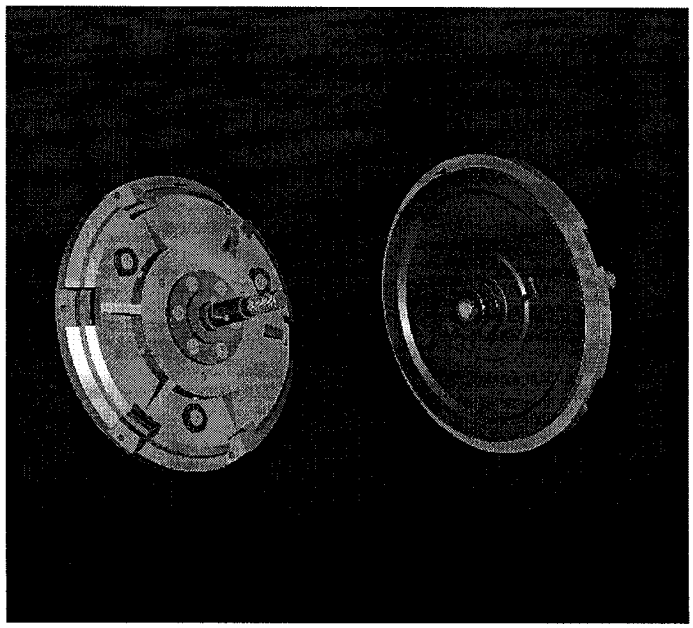


Figure 4-10(c) The Complete Motor

Figure 4- 10 The Overview of the Motorized Wheel

4.2.1 Constructing DSM Model 2

In this section, a DSM model is built based on the assembly relationship of all parts of the motorized wheel. The purpose of building this component-based DSM is to find an efficient plan to divide design tasks among each design team. The steps required to build the component-based DSM are listed as follows:

Step1 List all parts that the motorized wheel consists of in Table 4-2. The motorized wheel has 45 separate parts, but since several parts are not important for the design process and also since the software PSM32 is limited to 40 by 40 matrices, the DSM is simplified to include 40 parts only.

The principles of removing the five parts away are based on the "importance" and "relation". First, the removed parts cannot belong to the motor group and transmission group because the parts in the two groups are related to the core functions of the motorized wheel. Secondly, the removed parts cannot have an interdependent relation with other parts. Namely, the design processes of other parts do not demand information from them. For example, the sleeve is an accessory of the motorized wheel. The design of this part is sequential to the design of the transmission shaft and the hub of housing 2, but none of other parts demands information from it. Therefore, it can be chosen to be taken out.

Step 2 Correctly determining the relationships among 40 parts can be an onerous and difficult task; therefore, they are divided into four groups to form four small DSMs, which will then be combined into a larger DSM. The four groups are the same as the divisions in Table 2: transmission, motor, housing 1, and housing 2.

Step 3 The transmission group includes 12 parts, shown in Figure 4-11. A DSM, shown in Figure 4-12, is formed by these parts, and the relations among them are shown in the DSM. It is important that the relationship among parts be decided upon by all team members.

Table 4- 2 Separated Parts of the Motorized Wheel

Group		Part	Symbol
Transmission		shaft_trans	1
		bearing_4	2
		housing_trans	3
		driven_gear	4
		bearing_1	5
		counter_gear_2	6
		bearing_3 (*2)	7
		washer_ring_1	8
		stick_1	9
		counter_gear_1	10
		lock_key_2 (*2)	11
		elastic_ring	12
Motor		rotor	13
		bearing_5	14
		bearing_6	15
		lock_key_1	16
		cover_motor	17
		washer_2	18
		driving_gear	19
		circlip_1	20
		housing_motor	21
		electric_brush	22
		stick_2 (*3)	23
		support_frame	24
		magnet_block (*8)	25
		washer_3 (*3)	26
		screw_4 (*4)	27
		flank	28
		screw_3 (*6)	29
		bolt_1 (*6)	30
		nut_1 (*4)	31
Housing	Housing 1	housing_1	32
		hub_housing_1	33
		screw_1 (*6)	34
		circlip_2	35
		bearing_2	36
	Housing 2	housing_2	37
		hub_housing_2	38
		gear_ring	39
		screw_2 (*6)	40

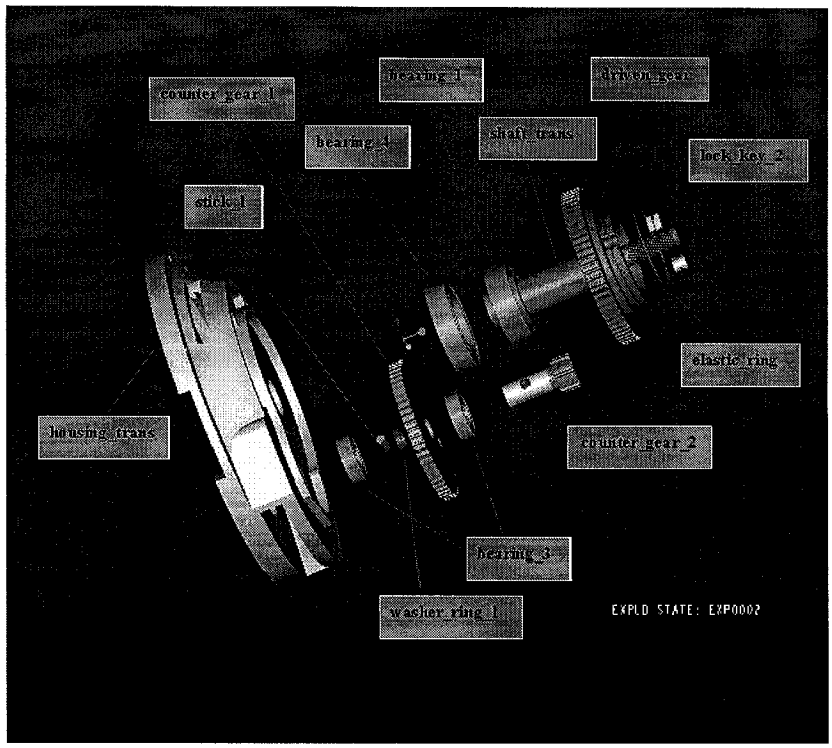


Figure 4- 11 The Exploded View of the Complete Transmission

	1	2	3	4	5	6	7	8	9	10	11	12
1 shaft_trans	*	0	0		0							
2 bearing_4		*	0									
3 housing_trans	0	0	*			0	0			0		
4 driven_gear		0		*	0	0						
5 bearing_1					*							
6 counter_gear_2			0	0		*	0			0		
7 bearing_3							*					
8 washer_ring_1						0		*		0		
9 stick_1						0			*	0		
10 counter_gear_1			0			0				*		
11 lock_key_2				0							*	
12 elastic_ring				0								*

Figure 4- 12 The DSM of Transmission Group

Step 4 The motor group consists of 19 parts, as shown in Figure 4-13. Based on the relations among these parts, a DSM is built, shown in Figure 4-14. It is a hard work for team members to correctly decide the relations of tasks in a 19*19 DSM.

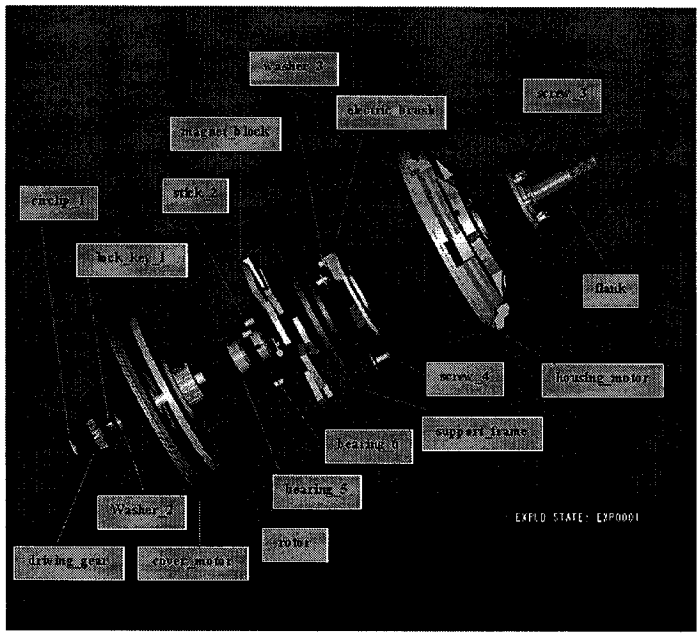


Figure 4- 13 The Exploded View of the Complete Motor

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1 rotor	*	0	0	0			0	0		0									
2 bearing_5		*																	
3 bearing_6			*																
4 lock_key_1				*			0												
5 cover_motor		0			*				0										
6 washer_2	0					*	0												
7 driving_gear	0						*												
8 circlip_1								*											
9 housing_motor			0		0				*	0	0	0			0	0	0	0	
10 electric_brush	0									*	0								
11 stick_2											*								
12 support_frame									0	0		*	0		0				
13 magnet_block												0	*						
14 washer_3														*	0				
15 screw_4															*				
16 flank									0							*			
17 screw_3																	*		
18 bolt_1																		*	
19 nut_1																		0	*

Figure 4- 14 The DSM of the Motor Group

Step 5 The next step requires building the DSM of Housing 1 with the five parts in Figure 4-15, and the DSM is shown in Figure 4-16.

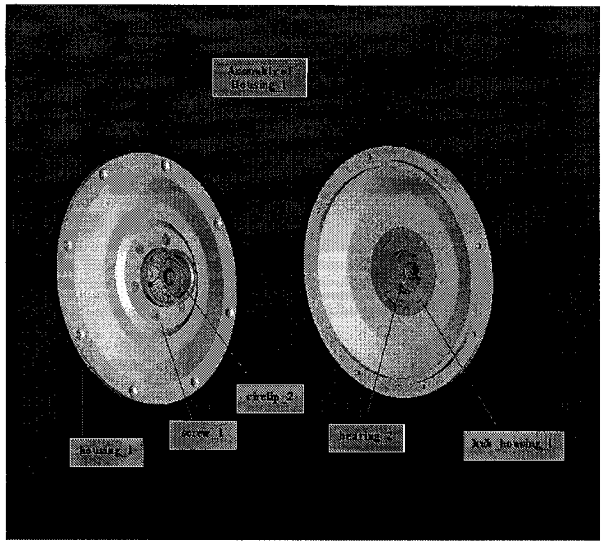


Figure 4- 15 The Complete Housing 1

	1	2	3	4	5
1 Housing_1	*	0	0		
2 Hub_Housing_1	0	*		0	0
3 screw_1			*		
4 circlip_2				*	
5 bearing_2					*

Figure 4- 16 The DSM of Housing 1 Group

Step 6 Using the four parts in Figure 4-17, the DSM of Housing 2 group is created, and the DSM is shown in Figure 4-18.

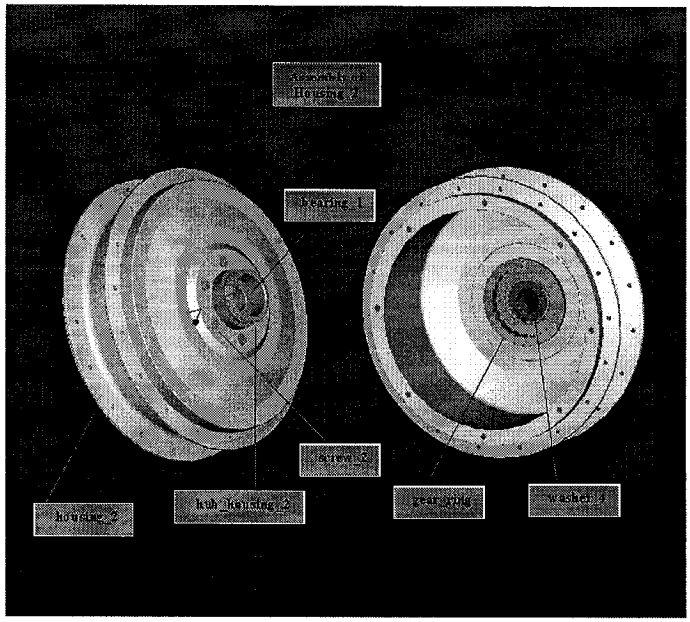


Figure 4- 17 The Complete Housing 2

	1	2	3	4
1 Housing_2	*	0	0	0
2 Hub_Housing_2	0	*	0	
3 Gear_Ring	0	0	*	
4 Screw_2				*

Figure 4- 18 The DSM of Housing 2 Group

Step 7 Finally, the four sub-DSMs in Figures 4-12, 4-14, 4-16, and 4-18 are gathered to form the DSM that represents the whole information of the design of the motorized wheel. At this point, the connections among the sub-DSMs have to be added to the matrix, as shown in Figure 4-20. Therefore, it is important to determine the following:

(1) In Figure 4-19, the relations between transmission housing (indicated as housing_trans in Figure 4-11) and motor housing (indicated as housing_motor in Figure 4-13).

(2) Adding the information flow between the gear ring and the lock key (indicated as gear_ring and lock_key_2 in Figure 4-11 and 4-17 respectively);

(3) The connection between bearing 1 and the hub housing (indicated as bearing_1 and hub_housing_2 in Figure 4-17);

(4) The relationship between housing 1 and housing 2 (indicated as housing_1 and housing_2 in Figure 4-10(a)).

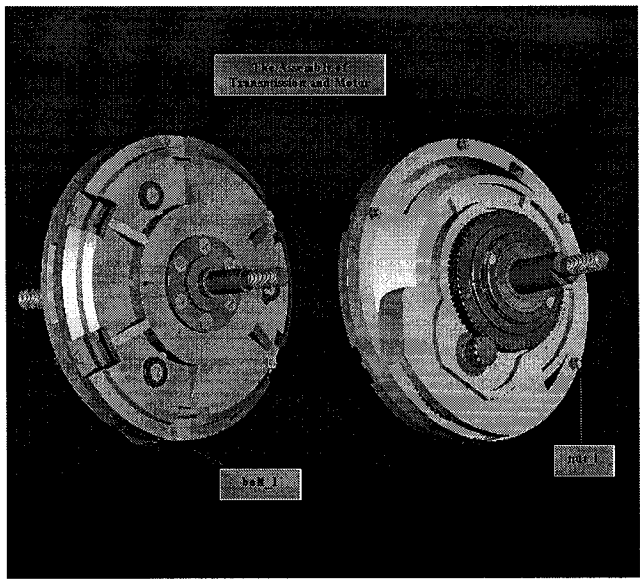


Figure 4- 19 The Assembly of Motor and Transmission

4.2.2 Partitioning and Tearing the DSM Model 2

In order to analyze the design process, the DSM in Figure 4-20 is partitioned first. The reordered DSM is shown in Figure 4-21. It can be found that iterations completely lie in blocks A and B along the diagonal. The size of block B is suitable, and it groups the main tasks of housing 1 and housing 2 into an iteration block. However, block A is too huge and complex because it includes all key components of the design work of the transmission and motor. Thus, this block needs to be reorganized. First, it is important to check the reason for which the transmission group and motor group work together. This question can be answered by studying the interrelated connections between transmission housing and motor housing. Therefore, an assumption can be made. Since the motor group is the most important assembly of the motorized wheel, the design of the motor housing can be done before the design of the transmission housing; in other words, the design of the motor housing does not need information from the design of transmission housing, but in contrast, the design of the transmission housing does depend on the motor housing.

Thus, the relations between the two parts in the original DSM are changed from two-way to one-way. Second, the new DSM must be partitioned, and this results in the matrix shown in Figure 4-22. In the new partitioned result, block B does not change its contents, block C consists of the components of the motor design, and block D focuses on the design of the transmission.

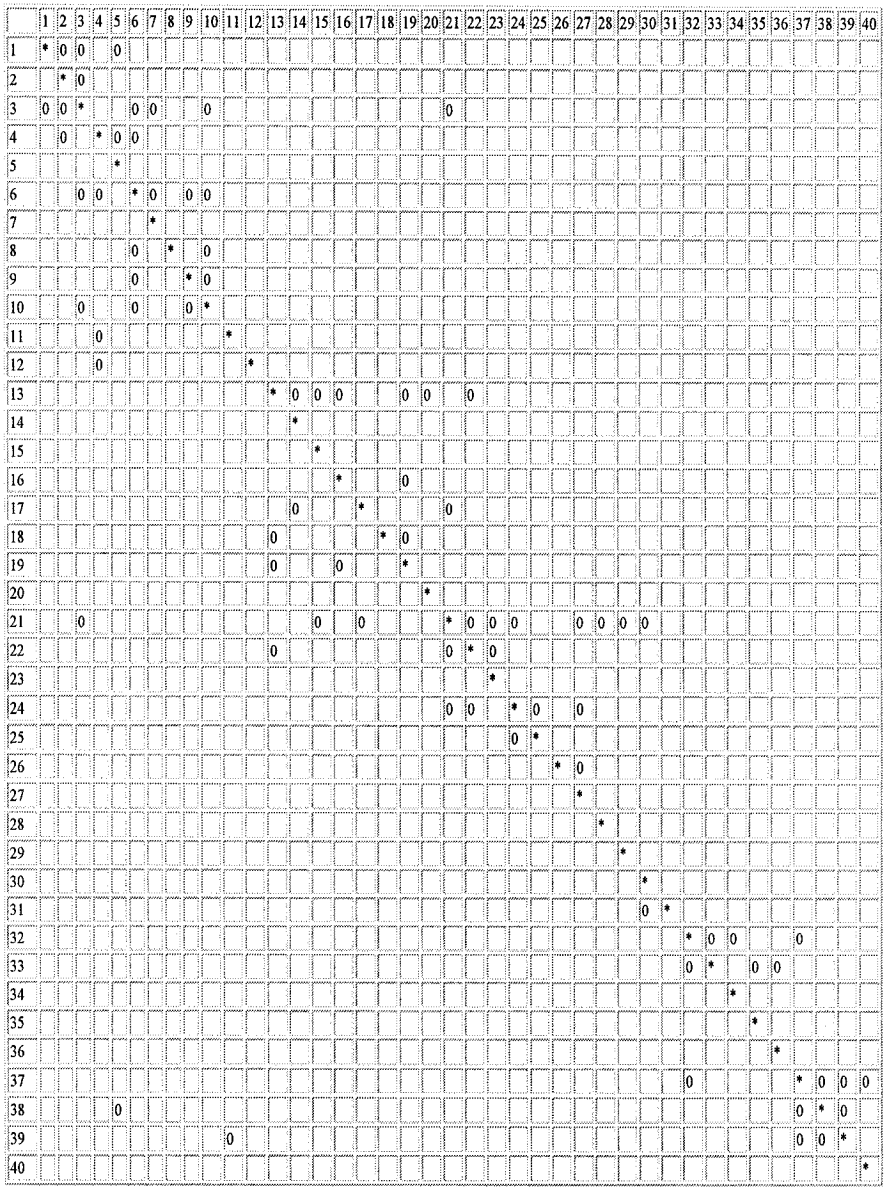


Figure 4- 20 DSM of the Design of the Motorized Wheel

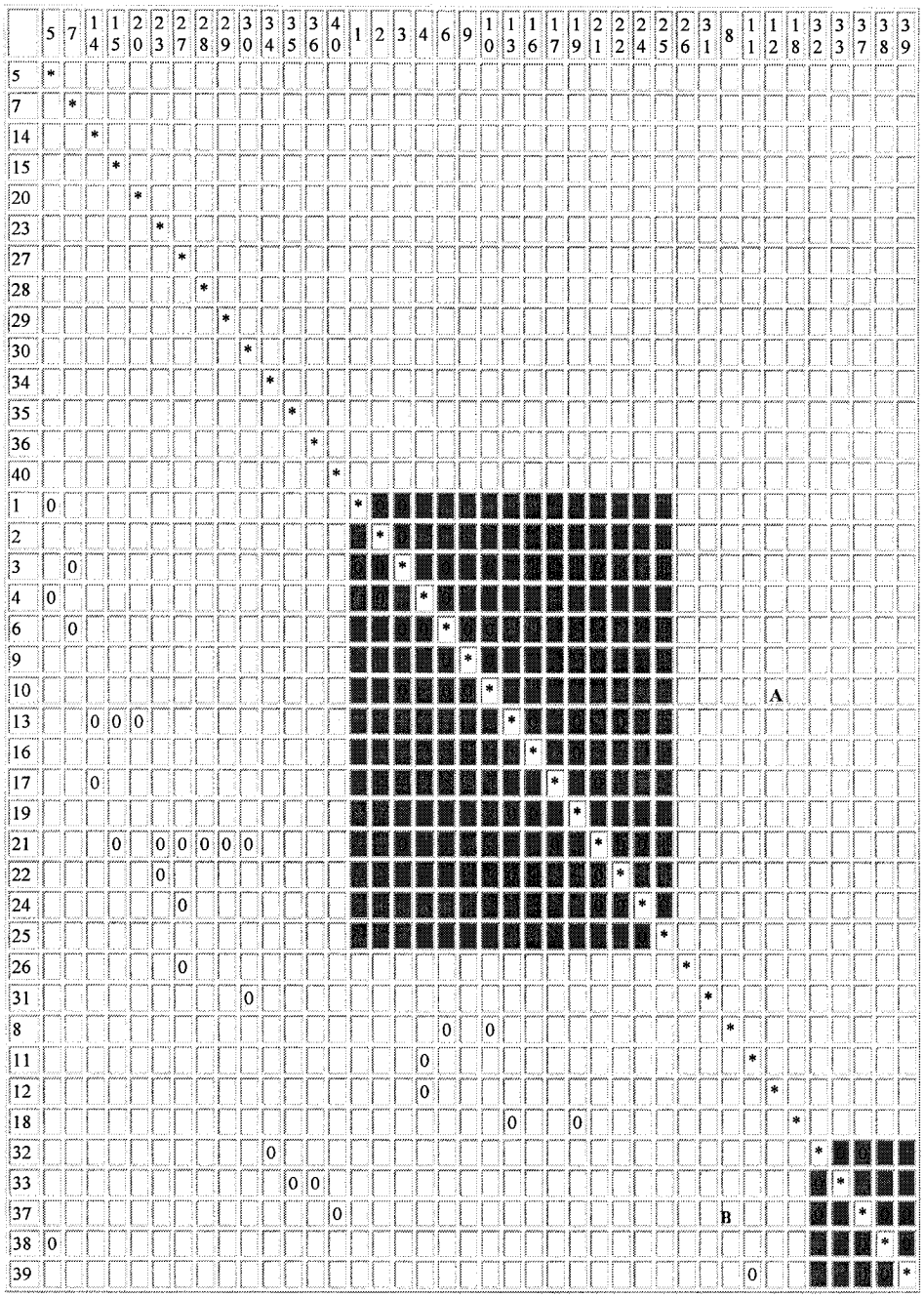


Figure 4- 21 The Partitioned DSM of the Original DSM

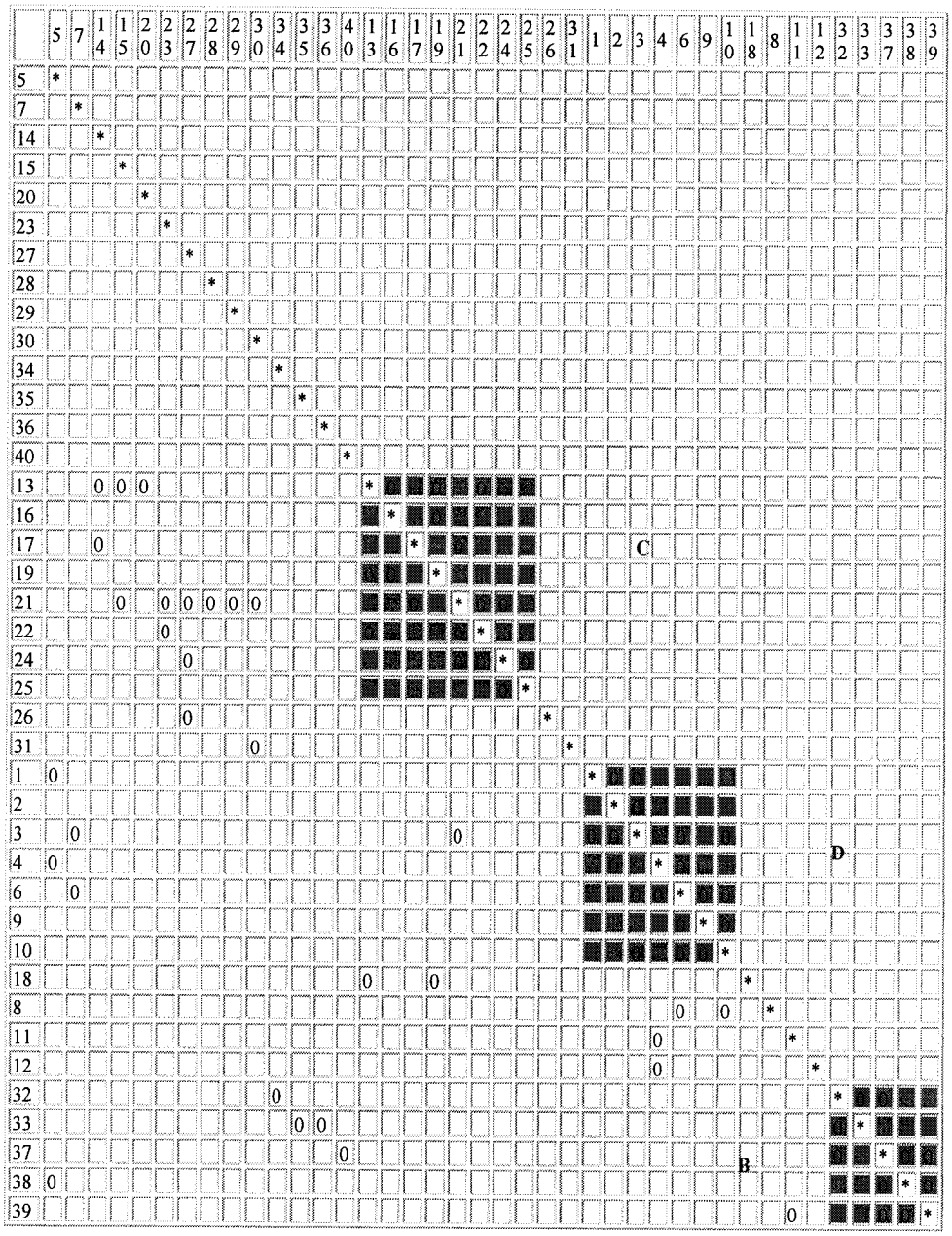


Figure 4- 22 The Partitioned DSM after the Rearrangement of the Relationship between Transmission Housing and Motor Housing

4.2.3 Division of Design Tasks

The DSM of Model 2 is a component-based DSM, and after partitioning, a cluster of blocks, B, C, and D, appear along the diagonal. Consequently, the design work can be divided into three design teams according to the blocks (other than the three blocks, the other components are mostly standard parts that can be found in the Machinery Handbook):

- Team 1: The design work on the motor group (Block C);
- Team 2: The design work on the transmission group (Block D);
- Team 3: The design work on the housing 1 and housing 2 groups (Block B).

4.3 Model 3 – Using DSM in the Design of a New Brake System

In this section, the DSM methodology is applied to analyze and arrange the design process of adding a new brake system to the motorized wheel. The DSM will be built as task-based DSM in order to generate an optimal plan to guide the design process of the brake system.

For the theoretical function of the brake system, when the motor is powered off, there is no current passing through coils of the brake system. At this time, the iron club fixed on the piston is pushed by the spring to work on the surface of the driven gear. Namely, the brake system prevents the wheel from running when the motor is in a static mode. When the motor is powered on, current through the coils generates a force to reverse the iron club back and hold it.

According to the structural space, the complete brake will be fixed in the space between the transmission housing and the motor cover as shown in Figure 4-23. The axial

length between the two housings is only 16.2 mm, so the structure of the brake system has to be arranged in a very small space.

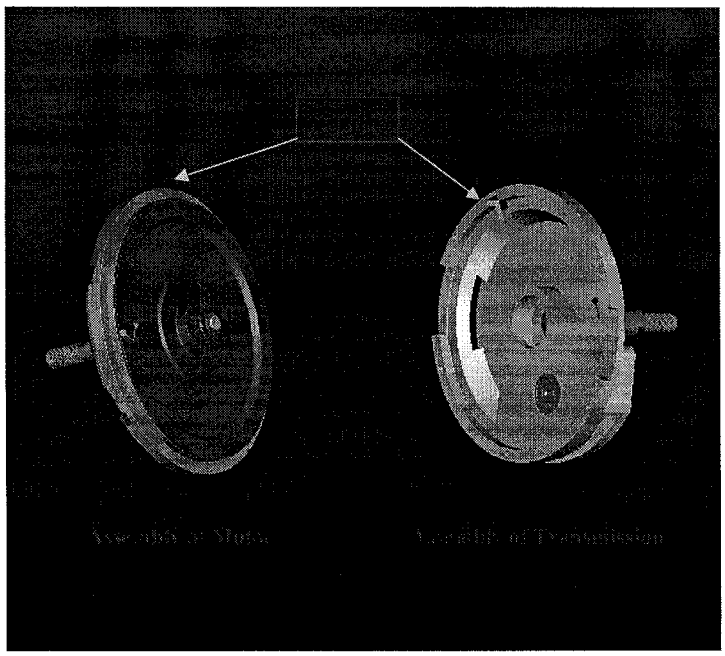


Figure 4- 23 The Brake Structure is fixed inside the Space between the two Assemblies

4.3.1 Construct the DSM of the Brake System

First, referring to some mechanical-electrical solenoids, the brake system is an eight-part electro-mechanical structure, as listed in Table 4-3. The brake system will be located in the space between the transmission housing and the motor cover, while the brake force works on the driven gear. Therefore, the relative parts from the original wheel are the transmission housing, motor cover, and driven gear.

Table 4- 3 The Brake System consists of 8 Parts

No.	Name	Usage
1	frame	Support all parts of the brake system
2	coils	Generate electrical force to reverse the iron club
3	spring	Generate the brake force
4	iron club	Brake the wheel on the surface of driven gear, and guide the spring
5	piston	Push the iron club
6	lock pin	Fix iron club and piston
7	bolts (*2)	Fasten the frame with the transmission housing
8	nuts (*2)	Work with bolts

The 8 parts in Table 3 and the relative three parts are gathered to form a DSM, as shown in Figure 4-24.

	1	2	3	4	5	6	7	8	9	10	11
1 design of frame	*		0		0		0		0	0	
2 design of coils	0	*									
3 design of spring			*		0						
4 design of iron club			0	*	0					0	0
5 design of piston			0	0	*	0					
6 design of lock pin				0	0	*					
7 design of bolt							*				
8 design of nut							0	*			
9 design of motor cover									*		
10 design of transmission housing										*	
11 design of driven gear											*

Figure 4- 24 The Original DSM for the Design of the Brake System

4.3.2 Partition and Tear the DSM to Analyze the Design Process

The software PSM32 can be used to help partition and tear the DSM. After partitioning, the DSM is changed into the form of Figure 4-25. An iteration block of four design tasks is located along the diagonal. The remaining design tasks, other than those in the block, can all be executed in parallel or in sequence. Thus, the key design work only focuses on

the execution of the block. The block can be torn at the piston design task. It is assumed that the piston design is known to the other three tasks in the block. After choosing tears, the block is completely torn as shown in Figure 4-26. The digit 3 is used to represent the output information from task 5 to the other three tasks.

	7	9	10	11	3	4	5	6	8	1	2
7 design of bolt	*										
9 design of motor cover		*									
10 design of transmission housing			*								
11 design of driven gear				*							
3 design of spring					*						
4 design of iron club			0	0		*					
5 design of piston							*				
6 design of lock pin								*			
8 design of nut	0								*		
1 design of frame	0	0	0		0		0			*	
2 design of coils										0	*

Figure 4- 25 The Partitioned DSM Equivalent to Figure 4-24

	7	9	10	11	3	4	6	5	8	1	2
7 design of bolt	*										
9 design of motor cover		*									
10 design of transmission housing			*								
11 design of driven gear				*							
3 design of spring					*			3			
4 design of iron club			0	0	0	*		3			
6 design of lock pin						0	*	3			
5 design of piston					0	0	0	*			
8 design of nut	0								*		
1 design of frame	0	0	0		0			0		*	
2 design of coils										0	*

Figure 4- 26 Tear the Block at the Design of Piston

The execution sequence of tasks can now be listed from the result of Figure 4-26:

(1) Tasks 9, 10, and 11 can be carried out in parallel.

(2) The block can be sequentially executed after task 10 and task 11. Inside the block, iterations are suggested to run in the following order: start with task 3, then task 4, then task 6, and finally task 5. The exact results of the four tasks will be reached after three or more iterations because task 5 is assumed to be known to other three tasks.

(3) Task 8 is parallel to the block because it only needs input from task 7.

(4) Task 1 can be completed sequentially when it receives information from task 8 and the block.

(5) Task 2 is sequential to task 1.

A flow chart is used to describe the above design process, as shown in Figure 4-27.

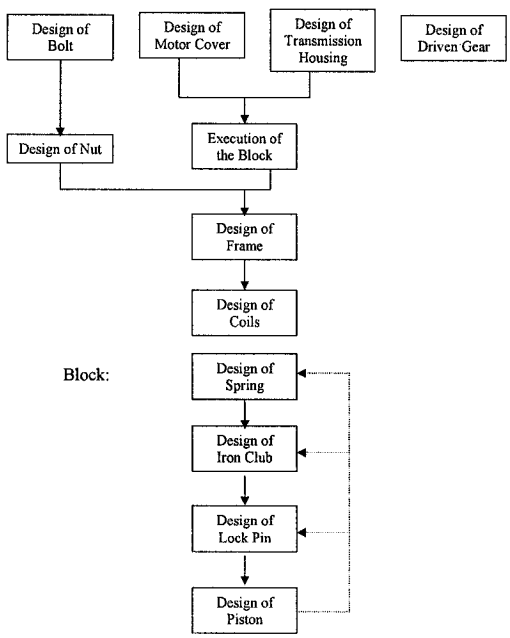


Figure 4- 27 The Flow Chart of the Design Process of the Brake System

4.3.3 Execution of the Design of the Brake System

The design process of the brake system can now be carried out according to the flow chart in Figure 4-27 and the information flow in Figure 4-26. Before executing this design, however, two aspects must be noted:

- The motor cover, transmission housing, and driven gear are not changed in the design process, and they are copied from the original parts to supply information to the design of other parts.
- In order to fix the brake system onto the motorized wheel, two holes have to be drilled on the transmission housing for bolts, and three grooves milled on the driven gear to fit the iron club. The two design tasks, drill holes and mill grooves, are not included in the matrix because they do not belong to the design of the brake system.

The design steps of the brake system are now discussed.

(1) Design of Bolt (Task 7)

The bolts are used to install the brake system on the motorized wheel. Their design is decided upon using the following information:

- The thickness of the transmission housing is 8.5 mm (from the design of the motorized wheel);
- The fringe thickness of the frame is assumed to be 5.5 mm.

According to ANSI/ASME B18.3.1M-1986, the bolts can be chosen as M3×0.5, as shown in Figure 4-28.

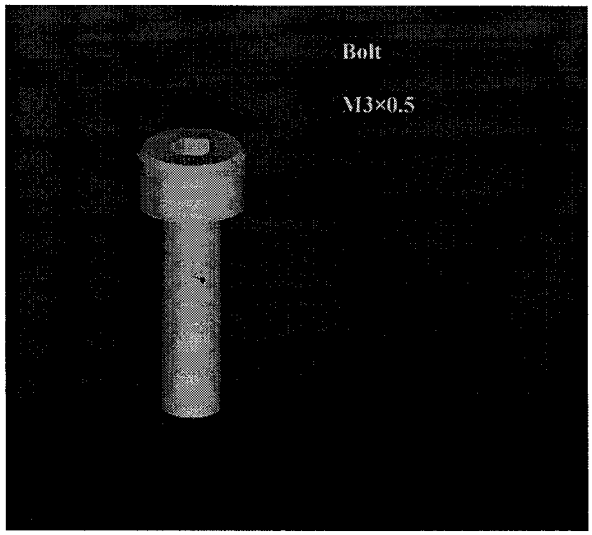


Figure 4- 28 Bolt

(2) Design of Nuts (Task 8)

To match the bolts that have been chosen, the nuts are also decided as M3×0.5, as shown in Figure 4-29.

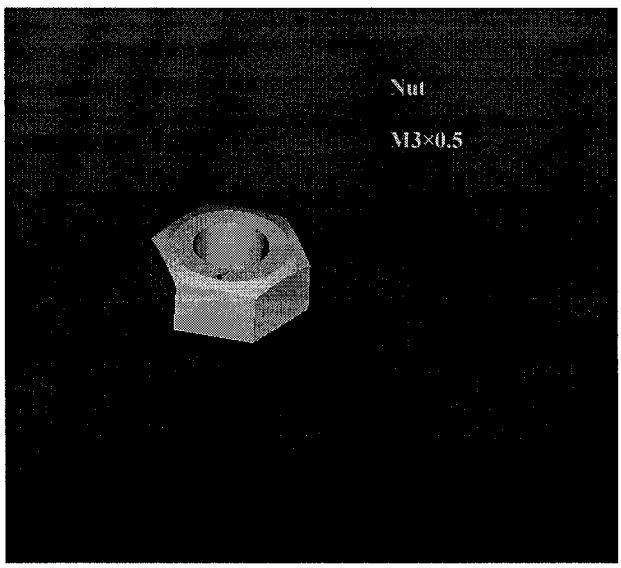


Figure 4- 29 Nut

(3) Design of Spring (Task 3)

The spring must be hard enough to just generate a force to press the iron club into the groove of the driven gear when the motor is out of electricity.

Assumption 1 The mean coil diameter is 6mm;

Assumption 2 The diameter of the wire is 1 mm;

Based on the above two assumptions, the outer diameter of the spring is 7 mm. There is no critical limitation for axial dimensions. But referring to the relationship between the spring and piston, a feedback from the piston will be possible to force the spring to change the above dimensions.

- The free length is 10mm, and it is pressed as length of 8mm in the assembly status;

The decision of the spring length has to be restricted by 16 mm, the whole height of the brake system.

- The pitch is 2 mm.

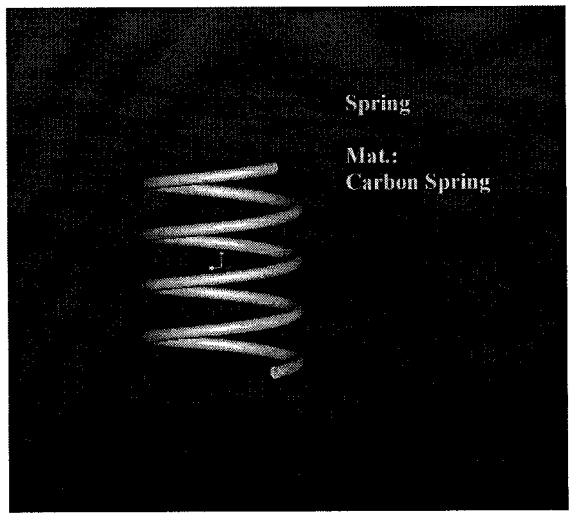


Figure 4- 30 Spring

(4) Design of the Iron Club (Task 4)

The diameter of the iron club should be big enough to hold the brake torque, and at the same time it cannot be bigger than the inner diameter of the spring (Φ6 mm). It is decided as being Φ4 mm. The length is up to the following axial dimensions:

- The thickness of the transmission housing: 8.5 mm;
- The gap between driven gear and transmission housing: 2.5 mm;
- The thickness of piston: 4 mm;
- The guidance for spring: 5 mm.
- The depth of grooves on the surface of driven gear is 3 mm.

Thus, the length of the iron club is 23 mm by adding all of the above dimensions. It needs to be confirmed by the feedback of the thickness of the piston, which means iterations take place between them. The hole that works with the lock pin is located 16 mm from one side, and its diameter is 1.5 mm.

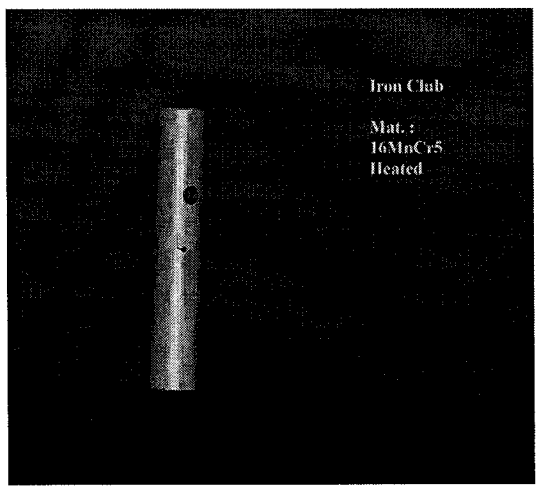


Figure 4- 31 Iron Club

(5) Design of Lock Pin (Task 6)

Matching the hole in the iron club, the diameter of the lock pin is also 1.5 mm, and its length is 7.8 mm. The outer diameter of the piston is assumed to be 8 mm.

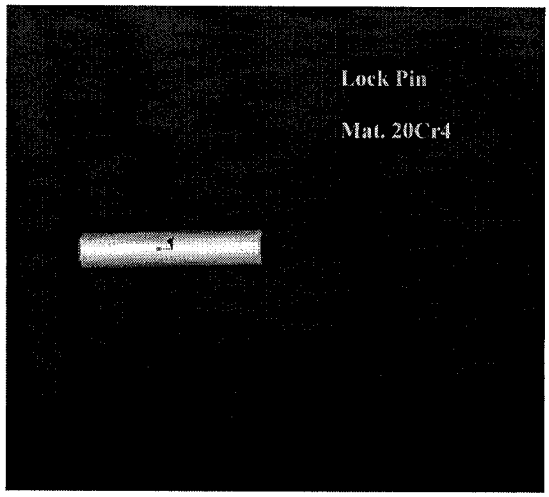


Figure 4- 32 Lock Pin

(6) Design of Piston (Task 5)

Referring to the assumption in the design of the lock pin, the outer diameter of the piston should be 8 mm, and inner diameter should be 4 mm. The thickness of the piston is 4 mm as the same assumption must be made in conjunction with the design of the iron club. The diameter of the hole that works with the lock pin is 1.5 mm, and this hole is located in the middle of the piston. Refer to Figure 4-33.

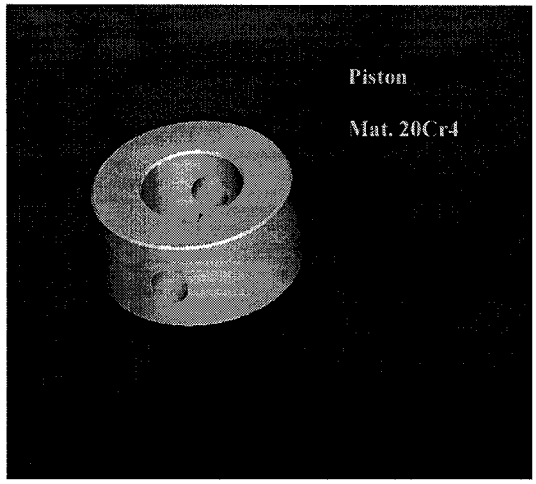


Figure 4- 33 Piston

(7) Design of Frame (Task 1)

The frame is the most complex part in the brake system, but it can be designed based on the results of the previous steps:

- Since the piston slides inside the hole of the frame, the diameter of the hole is decided as being 8 mm. The length of the hole should refer to the thickness of the piston (4 mm) and the free length of the spring (8 mm), so it is decided to be 12 mm.
- The thickness of the fringe is assumed to be 5.5 mm in the design of the bolt. Two counter bores are drilled on the fringe to fix the bolts: Φ5.5 mm x 3 mm-Φ3 mm x 2.5 mm.
- Restricted by the space between the motor cover and the transmission housing, the whole height of the frame is decided to be 16 mm.
- The length and width are not very important for this design, so they are decided as being 30 mm x15 mm, according to the space available.

- The upper side of the frame is designed to be conic to fit the generation of the electro-magnetic force.

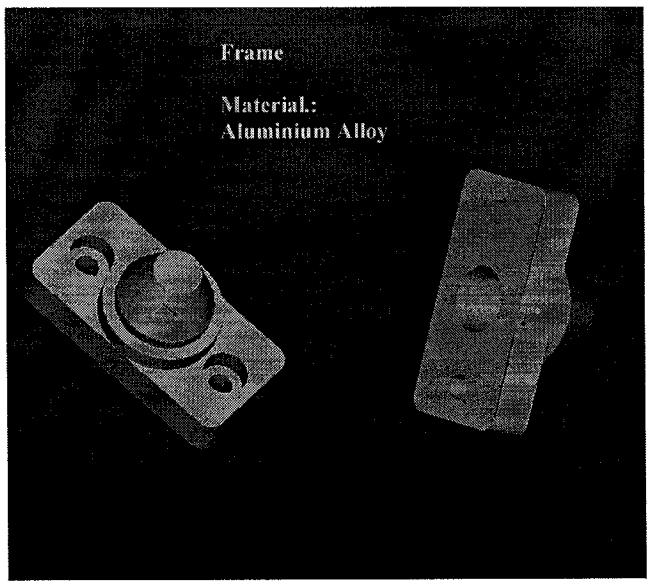


Figure 4- 34 Frame

(8) The Design of Coils (Task 2)

When the frame is finished, it is easy to roll coils to resemble the form of the frame, as shown in Figure 4-35.

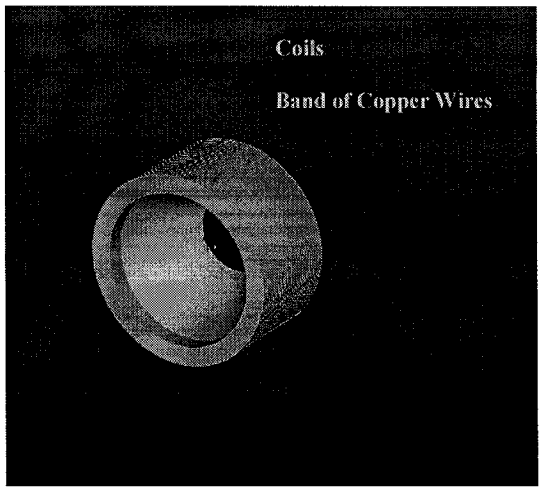


Figure 4- 35 Coils

(9) Design of the Assembly

The assembly of all parts that has been designed to form the complete brake system is shown in Figure 4-36.

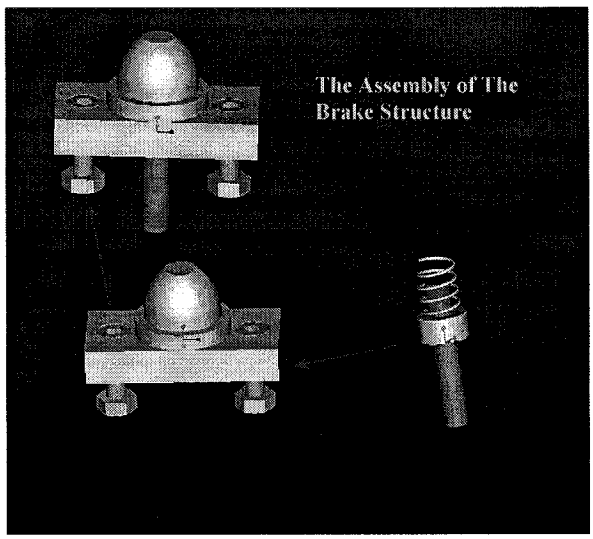


Figure 4- 36 The Complete Brake System

(10) Forward Design Work

While the work in this section does not belong to the DSM, it is relative and important to the brake system.

- Two counter bores are drilled in the transmission housing to fix the brake system to the motorized wheel, and an extra hole needs be drilled between the holes in order that the iron club can pass through it to work on the driven gear, as shown in Figure 4-37.

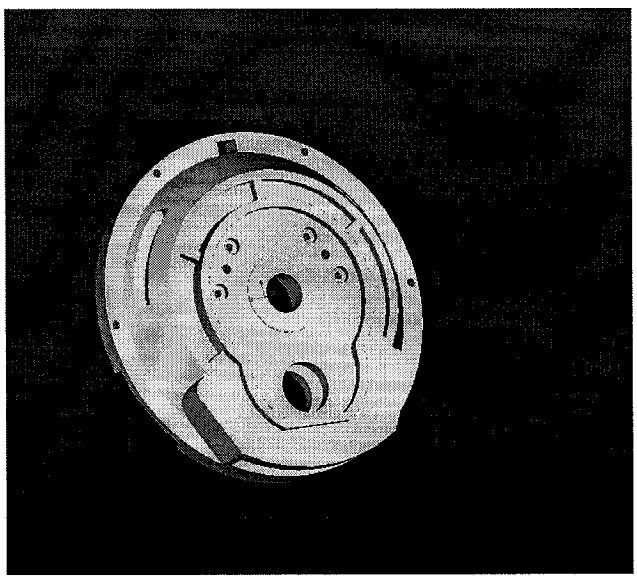


Figure 4- 37 Two Counter Bores and One Hole Drilled for Fixing One Brake Structure

- Three grooves are milled in the surface of the driven gear to hold the iron club, as shown in Figure 4-38.
- For the brake safety, two brake structures are fixed symmetrically in the motorized wheel, as shown in Figure 4-39.

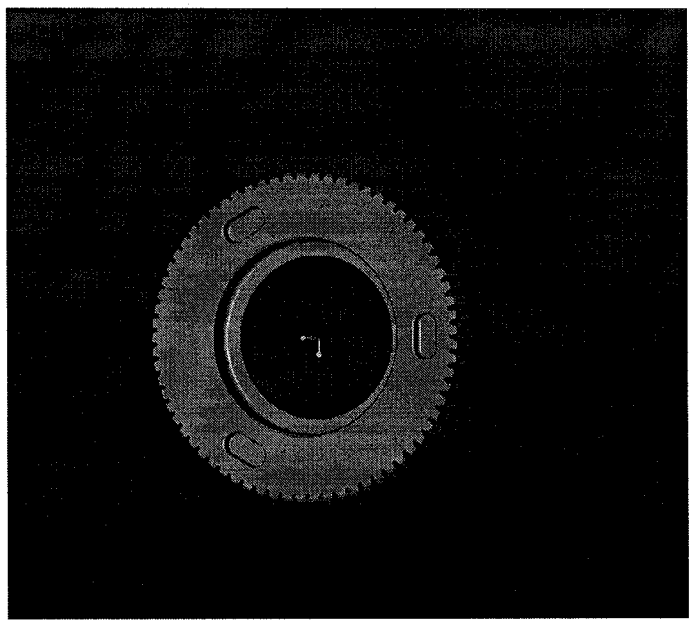


Figure 4- 38 Three Grooves Milled to Hold the Iron Club

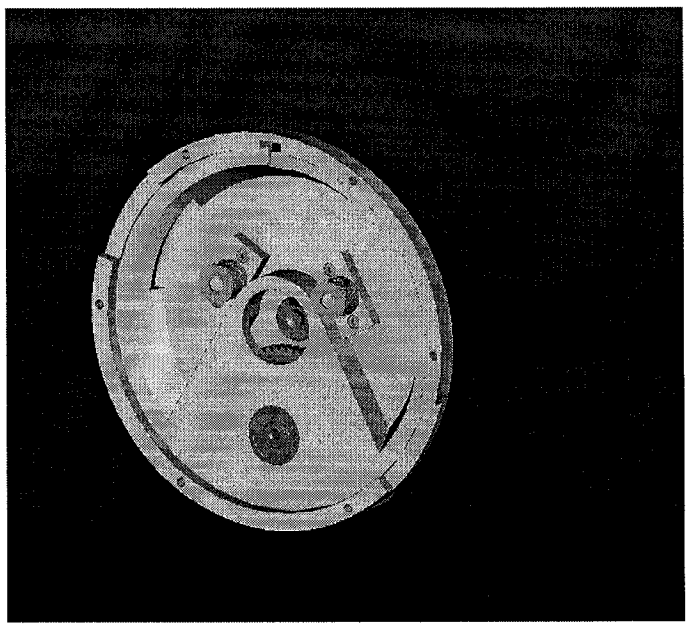


Figure 4- 39 Two Brake Structures Fixed to Ensure Brake Safety

4.4 Guidelines for Creating a Compact DSM

Often, the number of design tasks (or parameters) for a particular project is too large to fit into a manageable matrix. This may also be inconvenient in the use of a software, for example PSM32, which takes a matrix with a maximum size of 40 x 40. Recall that this was the case in the thesis for Model 2. Thus, when designers must develop a manageable DSM, i.e., one that does not result in an overly large matrix that is difficult to manipulate, it can be very useful to either reduce, eliminate, combine, or even leave out certain tasks in the analysis. Reducing the number of design tasks in the thought process results in reduced cost as well (Ulrich and Eppinger, 2004). In what follows, some guidelines for reducing a large matrix are presented.

- Importance of tasks/parameters: The evaluation of the importance of design tasks considers their significance relative to the main functions of product. For example, in Model 2, the components of the motor group and transmission group decide the main functions of the motorized wheel. These components cannot be ignored. However, some parts that are part of the housing of these groups are not important in the design process in comparison to other parts; hence their elimination does not influence the main functions of the product. Thus, they can be removed without any significant impact on the end result. Care must be taken to ensure that such tasks can indeed be considered as being less important by obtaining consensus among team members.
- Relations among tasks/parameters: Design tasks that have little or no relation to other tasks in the matrix can be eliminated because of their independence. Thus tasks that are interdependent or sequential should not be eliminated from the matrix. For

example, the sleeve on the transmission shaft is not relative to the design of other parts, so it can be removed.

- Combination of tasks/parameters: Some design tasks are worth considering together, i.e., in combination, to reduce the total number of components. For example, some components that can use the same material and can be fused functionally, and can thus be considered to be designed as a single part.
- Evaluation matrices: It is difficult to choose which design tasks to remove when many are involved in a process. Evaluation matrices are helpful for designers to make such decisions (Hyman B., 2003). The evaluation matrices should include as many criteria as possible to help designers prioritize and select which tasks/parameters are the most important to consider in the DSM. In the development of an evaluation matrix, various criteria can be considered. Weighted values can be assigned to various criteria to help in the selection of final task/parameters.
- Simplification of tasks: Simplification of design tasks can help to remove tasks or parameters that are unnecessary or not useful for the customer, and can help reduce the time needed to manufacture the part, as well as reduce material needs. Design simplification can help designers devote their time for critical activities. By simplifying the design, a more manageable DSM can be obtained.
- Design for Manufacturing/Assembly (DFM/DFA): DFM/DFA is a methodology that addresses costs to link customer needs and product specifications (Ulrich and Eppinger, 2004). According to this methodology, the reduction of the cost of components and assemblies should be considered in the design process. In order to reduce the cost of components, standard parts are suggested to be used as much as

possible. The dimensions of standard parts, for example, can be easily determined based on machining handbooks, which greatly reduces the total design time. Standard parts are available for purchase on the market so that the future manufacturing cost can be saved. The assemblies always incur labor cost and equipment cost, so in the design process, the integration method is necessarily taken for those parts that can be combined theoretically. For example, molded or forged parts are suggested to be used in the design process to reduce the number of components and subassemblies. Thus, the principles of DFM/DFA can be useful in reducing the size of the DSM.

The principles outlined above were useful in the construction of the DSM in this thesis. For more complex products, these guidelines should be followed carefully.

4.5 Conclusion

In the three models, the DSM is used as a management tool to organize the design processes. Most examples of DSM only describe its application for arranging large design processes at a general level. Through model 1, it can be found that DSM also can also be used for detailed design, for example dimensional decisions. Model 2 is a comparatively large design process. It is an example of using the DSM to organize a complex design process. Model 3 involves a design process to create a new structure. In each model, partitioning is used to rearrange the task sequence, and tearing is used to help find the best way to break the coupled tasks by making less assumptions. It is important to note that tearing the block is only used to find a way to execute coupled tasks in sequence or parallel, and when they are carried out they are restricted by these assumptions. After partitioning and tearing, the DSM proposes an execution process.

CHAPTER 5 NUMERICAL EVALUATION OF DSM

In this chapter, the application of the numerical DSM is discussed. The shortest design time is calculated for the coupled tasks in Model 3. The contribution of each coupled task to the convergence of the design process is also calculated to reveal the numerical evaluation of the DSM.

5.1 Calculation of the Minimal Execution Time

From the production flow chart shown in Figure 4-27, shown again below, the design process of the brake system (model 3) suggests eleven tasks.

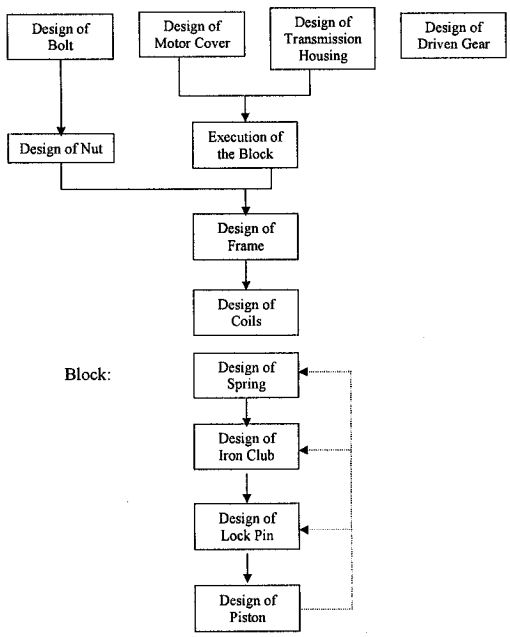


Figure 4-27 The Production Flow Chart of the Design Process of the Brake System

The direction of the information flows of these tasks can be identified from this figure. For those tasks which are in parallel, the design time is simply equal to the

maximum task time among all tasks. For sequential tasks, the design time is calculated by adding the individual task times together. Thus, the total design time of the brake structure can be calculated by using the following formula:

Total Design Time = Max{task 7, task 9, task 10, & task 11} + max{task 8, f(block)} + task 1 + task 2 (5.1)

where f(block) is the design time of coupled tasks in the block.

If the time for every design task can be estimated, only the function f(block) is unknown in Formula 5.1. Therefore, the length of the design time of the block decides the total design time of the design process.

The block in Figure 4-27 is taken out to consider the function f(block), as shown in Figure 5-1. Several steps are needed to transfer this block into a form that is suitable for calculation. For convenience, tasks 3, 4, 5, and 6 are referred to as A, B, C, and D. First, the time to complete each task alone (without considering the influence of other tasks) is predicted and put along the diagonal of the matrix. Next, the term a_{ij}, the probability that the i_{th} task requires rework due to the j_{th} task, replaces the 0s in the off-diagonal blanks. The term a_{ij} is between 0 and 1, and $i \neq j$. The empty blanks are filled with 0s because of no rework requirements between those tasks. The new matrix is shown in Figure 5-2. The values determined in step 2 and step 3 should be based on the discussion and input of all team members.

	3	4	5	6
3 Design of spring	*		0	
4 Design of iron club	0	*	0	
5 Design of piston	0	0	*	0
6 Design of lock pin		0	0	*

Figure 5- 1 The Block Taken out of Figure 4-47

	A	B	C	D
A	3	0	0.1	0
B	0.1	7	0.3	0
C	0.1	0.2	5	0.1
D	0	0.1	0.1	2

Figure 5- 2 Probabilities of Rework and Task Execution Time

There are 4! = 24 possible execution sequences among the four tasks A, B, C, and D. The sequence A-B-C-D is chosen to be an example to discuss the calculation process. According to the possibilities in Figure 5-2, a Markov chain is built to express the iteration process, as shown in Figure 5-3.

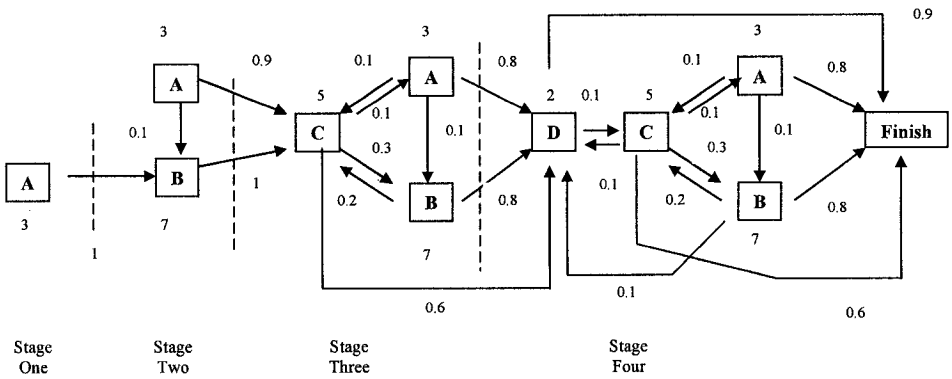


Figure 5- 3 Markov Chain expressing the completing Probabilities for ABCD

In Figure 5-3, the execution process is divided into four stages. From Stage 1 to Stage 2, tasks A and B are carried out sequentially. In Stage 1, the design work starts at task A; in Stage 2, task B is carried out first, so iterations take place between task A and task B until they converge. Similarly, Stage 3 starts at task C, and Stage 4 starts at task D. The expected time remaining is used to describe the execution time to finish each task. It is the sum of the task time itself plus the probability of reworking the other tasks. Since task D begins in Stage 4, the expected time remaining of task D (r_D) is captured in this stage; similarly, the expected time remaining of task C (s_C) is obtained in Stage 3; for task B, the time remaining (t_B) is found in Stage 2; finally, the time remaining of task A (u_A) is found in Stage 1. The calculation steps are introduced as follows.

(1) Calculation of the expected time remaining r_D in Stage 4.

The symbols r_A, r_B, r_C, and r_D represent the time remaining at each node A, B, C, and D in Stage 4. The linear equations are as follows:

$$\begin{aligned} r_A &= 0.1\, r_B + 0.1\, r_C + 3 \\ r_B &= 0.2\, r_C + 0.1\, r_D + 7 \\ r_C &= 0.1\, r_A + 0.3\, r_B + 0.1\, r_D + 5 \\ r_D &= 0.1\, r_C + 2 \end{aligned} \tag{5.2}$$

The linear equations in 5.2 represent the relationships of the four tasks in Stage 4. The iterations among them are carried out according to 5.2 until the design work has converged. The equations in 5.2 can be written in matrix form:

$$\begin{bmatrix} 1 & -0.1 & -0.1 & 0 \\ 0 & 1 & -0.2 & -0.1 \\ -0.1 & -0.3 & 1 & -0.1 \\ 0 & 0 & -0.1 & 1 \end{bmatrix} \begin{bmatrix} r_A \\ r_B \\ r_C \\ r_D \end{bmatrix} = \begin{bmatrix} 3 \\ 7 \\ 5 \\ 2 \end{bmatrix} \tag{5.3}$$

Using Gaussian Elimination 5.3:

$$\begin{bmatrix} 1 & -0.1 & -0.1 & 0 \\ 0 & 1 & -0.2 & -0.1 \\ 0 & 0 & 0.928 & -0.131 \\ 0 & 0 & 0 & 0.986 \end{bmatrix} \begin{bmatrix} r_A \\ r_B \\ r_C \\ r_D \end{bmatrix} = \begin{bmatrix} 3 \\ 7 \\ 7.47 \\ 2.805 \end{bmatrix} \tag{5.4}$$

Thus,

$r_D = 2.845$

r_A, r_B, and r_C do not need to be calculated because only task D starts in Stage 4.

(2) Calculation of the expect time remaining s_C in Stage 3.

The terms s_A, s_B, and s_C represent the time remaining at each node A, B, and C in stage three. Again, the linear equations are as follows:

$$\begin{aligned} s_A &= 0.1\,s_B + 0.1\,s_C + 3 \\ s_B &= 0.2\,s_C + 7 \\ s_C &= 0.1\,s_A + 0.3\,s_B + 5 \end{aligned} \tag{5.5}$$

Writing these equations in a matrix form gives:

$$\begin{bmatrix} 1 & -0.1 & -0.1 \\ 0 & 1 & -0.2 \\ -0.1 & -0.3 & 1 \end{bmatrix} \begin{bmatrix} s_A \\ s_B \\ s_C \end{bmatrix} = \begin{bmatrix} 3 \\ 7 \\ 5 \end{bmatrix} \tag{5.6}$$

Again, Gaussian Elimination results in:

$$\begin{bmatrix} 1 & -0.1 & -0.1 \\ 0 & 1 & -0.2 \\ 0 & 0 & 0.928 \end{bmatrix} \begin{bmatrix} s_A \\ s_B \\ s_C \end{bmatrix} = \begin{bmatrix} 3 \\ 7 \\ 7.47 \end{bmatrix} \tag{5.7}$$

Thus,

$s_C = 8.050$

(3) Similarly, t_A and t_B refer to the remaining time in Stage 2, and using the same procedure as before:

$$t_A = 0.1\, t_B + 3 \tag{5.8}$$

$t_B = 7$

(4) Finally, u_A, the remaining time in stage 1, is:

$u_A = 3$

(5) The total execution time of the function f(block) is the sum of the time remaining in each stage:

$$r_D + s_C + t_B + u_A = 2.845 + 8.050 + 7 + 3 = 20.895 \tag{5.9}$$

The above calculations were done for sequence ABCD. For every execution sequence, a new Markov chain must be created to reflect the changes in task execution sequence. The calculation of the sequence ABDC is also presented in detail in order to allow the reader to completely understand the changes in calculations for changes in task sequence.

The Markov chain that expresses the probabilities of completing each task is shown in Figure 5-4. Again, observing the probabilities in the Markov chain, task A can be successfully completed (100%) in the first stage, as can task B in the second stage, and

task D in the third stage. Thus, the expected time remaining for each task in the first three stages is as follows:

$s_D = 2$

$t_B = 7$

$u_A = 3$

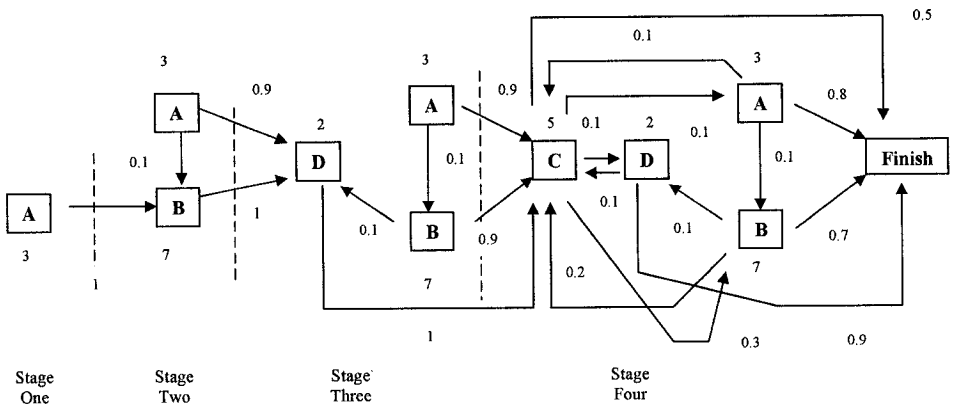


Figure 5- 4 Markov Chain expressing the completing Probabilities for ABDC

Now only the expected time remaining in the fourth stage, r_C , is unknown. The formulas based on the expected time remaining are as same as those in 5.2. According to these equations, a new matrix is built to calculate the expected time remaining of task C in this stage.

$$\begin{bmatrix} 1 & -0.1 & 0 & -0.1 \\ 0 & 1 & -0.1 & -0.2 \\ 0 & 0 & 1 & -0.1 \\ -0.1 & -0.3 & -0.1 & 1 \end{bmatrix} \begin{bmatrix} r_A \\ r_B \\ r_D \\ r_C \end{bmatrix} = \begin{bmatrix} 3 \\ 7 \\ 2 \\ 5 \end{bmatrix} \tag{5.10}$$

Applying the same calculations as in the previous task sequence, $r_C = 8.451$

The total execution time of sequence ABDC is:

$$r_C + s_D + t_B + u_A = 8.451 + 2 + 7 + 3 = 20.451$$

Similar to the calculation of the above two sequences, the remaining 22 execution sequences can be calculated. The results are shown in Table 5-1.

Table 5- 1 The Expected Time of all Possible Execution Sequences of Tasks in the Block

1st	2nd	3rd	4th	The Expect Time (Time Unit)
A	B	C	D	20.895
A	B	D	C	20.451
A	C	B	D	19.812
A	C	D	B	19.893
A	D	C	B	**19.591**
A	D	B	C	20.651
B	A	C	D	21.595
B	A	D	C	21.151
B	C	A	D	21.257
B	C	D	A	**22.090**
B	D	A	C	21.171
B	D	C	A	21.683
C	A	B	D	19.993
C	A	D	B	20.074
C	B	A	D	20.214
C	B	D	A	21.048
C	D	A	B	20.608
C	D	B	A	21.139
D	A	B	C	20.651
D	A	C	B	**19.591**
D	B	A	C	21.371
D	B	C	A	21.883
D	C	A	B	19.796
D	C	B	A	20.867

From the result in Table 5-1, the execution time of iteration tasks is different for different execution orders. The minimal execution time is 19.591 from the sequence ADCB or DACB, and the longest execution time is 22.090 from the sequence BCDA. The execution sequence BCAD takes 12.76% more design time than the sequences

ADCB and DACB. Consequently, a correct choice of the execution can obviously save the total design time.

In Figure 4-27, the recommended execution sequence after tearing the block is: design of spring, design of the iron club, design of lock pin, and design of piston. This refers to the order ABDC. This result is different from the optimal sequence in Table 5-1. The reason that causes the difference is that the strength of probabilities is not considered when the software PSM32 tears the block in Figure 4-26. Namely, PSM32 regards all 0s as the same strength of interdependencies. Thus, the calculation of the shortest design time above can be taken as a new tearing method to break coupled tasks. Compared to the method of the longest circuit in PSM32 (refer to the Appendix), this method tears DSM in a quantitative way.

5.2 Calculation of the Total Eigenvector

In this section, the eigenvalues, eigenvectors, and total eigenvectors are calculated and used to describe the contribution of coupled tasks to the convergence of the design process (refer to Section 3.4.4). If a task contributes more to the convergence of the design process than other tasks, it takes more design time than others, so it is very meaningful to study the effect of this task on the total design time. The calculation is also based on the rework probabilities among tasks in Figure 5-2. All rework probabilities are put in a matrix A, and the calculation is extended from the matrix A.

$$A = \begin{bmatrix} 0 & 0 & 0.1 & 0 \\ 0.1 & 0 & 0.3 & 0 \\ 0.1 & 0.2 & 0 & 0.1 \\ 0 & 0.1 & 0.1 & 0 \end{bmatrix}$$

Decomposing the matrix A, its eigenvectors matrix S and eigenvalues matrix Λ can be calculated as follows:

$$S = \begin{bmatrix} 0.1961 & -0.1724-0.2131i & -0.1724+0.2131i & -0.2683 \\ 0.6512 & -0.3694+0.1080i & -0.3694-0.1080i & -0.6967 \\ 0.6110 & 0.0906+0.0389i & 0.0906-0.0389i & 0.6652 \\ 0.4051 & 0.8758 & 0.8758 & 0.0127 \end{bmatrix}$$

$$\Lambda = \begin{bmatrix} 0.3116 & 0 & 0 & 0 \\ 0 & -0.0318+0.0168i & 0 & 0 \\ 0 & 0 & -0.0318-0.0168i & 0 \\ 0 & 0 & 0 & -0.2479 \end{bmatrix}$$

The columns of S refer to the eigenvectors, and the elements in the diagonal of Λ refer to the eigenvalues that match the eigenvectors in S. The three matrices have the relationship:

$$A = S\Lambda S^{-1} \tag{5-10}$$

In the matrix S, the sequence of the four tasks that contribute to the convergence of the design process is BCDA (from the most contribution to the least contribution). At the same time, the calculation of the first four work vectors also gives the same conclusion, shown below. The values of the work vectors are ranked from task B, task C, task D to task A, respectively.

$$u_0 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad u_1 = \begin{bmatrix} 0.1 \\ 0.4 \\ 0.4 \\ 0.2 \end{bmatrix} \quad u_2 = \begin{bmatrix} 0.04 \\ 0.13 \\ 0.11 \\ 0.08 \end{bmatrix} \quad u_3 = \begin{bmatrix} 0.011 \\ 0.037 \\ 0.038 \\ 0.024 \end{bmatrix}$$

Finally, the calculation of the total work vector can verify the above conclusion of the contribution sequence to the total design work:

$$U = S(I-\Lambda)^{-1}S^{-1}u_0 = \begin{bmatrix} 1.1564 \\ 1.5849 \\ 1.5641 \\ 1.3149 \end{bmatrix}$$

Sequencing the four tasks according to the magnitudes of their total work vectors gives BCDA. In Table 5-1, the execution sequence BCDA consumes the longest design time. Thus, for the purpose of reducing design time, the execution of coupled tasks should start from the task that contributes least to the convergence of the total design work; namely, the design work should begin from these single tasks to complex ones. In Table 5-1, the sequences ADCB and DACB all support this conclusion.

5.3 Sensitivity Analysis

Because the DSM is populated with data which are based on team members' assessment of the design process, the results of the DSM are highly dependent on the subjective inputs of teams. While as many people as possible should be consulted in the building of the matrix, it is important to verify the effect of changing the input parameters. A sensitivity analysis was conducted by changing the dependency strengths in the DSM, and results show that changes in results but changing inputs are not significant. Details of the analysis can be found in Appendix B.

5.4 Conclusion

The calculation of the shortest design time and contribution to the convergence of the design process are two numerical methods used to solve the coupled design tasks mathematically. The calculation processes are based on the rework probabilities. The results of the two methods have the same conclusion. Either of the two methods can therefore be useful for design managers and designers.

The method of the shortest design time uses a Markov chain to build mathematical equations that represent the relationships among coupled tasks. These tasks are carried out in iterations until they are finished. Different execution orders have different design times. In the end, the shortest design time can be determined from these calculation results.

The calculation of the convergence to the design process uses a work vector and total eigenvector to find which tasks takes the longest design time among these coupled tasks. The work vector in every iteration gives the remaining percentage of unfinished work. The remaining percentages of task B and task C are greater than other tasks, so they contribute more to the convergence of the design process; namely, they take more design time. The total eigenvectors gives the numerical estimation to the total amount of work. The result is more exact. B turns out to be the largest one, so it contributes most to the convergence of the design process.

The calculation of the shortest design time and contribution to the convergence of the design process evaluate and estimate the design process mathematically.

CHAPTER 6 CONCLUSIONS

This thesis has introduced the application of the Design Structure Matrix method in modeling the product design process. The design of the motorized wheel is used as a practical application to describe the DSM method in detail. The discussion focuses on the two powerful functions of the DSM in the product development process: efficient reordering of design tasks and quantitative evaluation of the design process.

This thesis discussed the reordering methodology of the DSM. The proper ordering of design tasks can help to avoid unnecessary time consuming and costly rework in the product development process. After reordering of the matrix, which tasks should be done sequentially or in parallel, and which ones should be executed concurrently are obvious. Coupled tasks form one or more blocks along the diagonal of the DSM, thus making it easy for managers and designers to determine the design sequence from the reordered matrix. At the same time, the reordering method can reveal the weaknesses in the organization. Unplanned tasks, which cannot be coupled in blocks, force managers and designers to go back to their initial design models to reorganize the design contents and rebuild the relationships of tasks.

In this thesis, the reordering method has been applied in three design models. Through the application, the solution of dimensional decisions, the design of the motorized wheel, and the new brake system design have been identified. Based on the results obtained from these three models, it can be concluded that the DSM can be very useful for managers and designers in organizing the sequence of the design process and in identifying task divisions.

Two numerical methods of DSM were used to model and calculate iterative design tasks. The most difficult work in a design process is the understanding of how to treat rework relations among tasks. The two methods presented in this thesis demonstrate the mathematical algorithms that can consider this problem through the use of rework probabilities. The first method determines the shortest design time based on rework. Once the shortest design time of iterative tasks is found, an efficient design sequence is also found because the other tasks are sequential or parallel. The second method determines the contribution of coupled tasks to the speed of convergence of the design process. The contribution to convergence can also give the policy to reduce design time: namely, executing iterative design tasks from the least contribution to the convergence of the design process to the most contribution. The design tasks which contribute more to the convergence take more design time. Thus, the rework of these tasks also consumes more time. If they were carried out first, more rework would take place. Therefore, the tasks of less contribution to the convergence are expected to be executed before these tasks. The result based on this conclusion has been compared with the calculation of the shortest design time. For instance, the two methods applied in the design time calculation of the design of the new brake system have the same conclusion. The shortest design time is important because it not only means reducing product development time but also reducing design cost as a result. The above two kinds of calculations help engineering managers and designers to make better decisions in the organization of their design process.

The advantages of DSM can be presented in three aspects. First, the DSM uses a matrix to compact all information of product development in a concise form. Especially

for an iterative design process, it is difficult to model it in other ways. For instance, for the practical applications presented in this thesis, the matrices used in the decision of axial dimensions, the design of the motorized wheel, and the design of the new brake system all gather design tasks and their relationships (input and output) in information pools. As a management tool, the DSM is simple and efficient in the presentation of the product development process. Second, the reordering function of the DSM helps managers and designers to reorder the design process according to the relationships of design tasks. The reordering process results in the generation of an execution sequence by which tasks can be scheduled. Third, the numerical DSM can not only evaluate the design process quantitatively, but also continues to improve the design process. For example, the calculation of the iteration block in the brake system design in this thesis gave a quantitative evaluation for the different execution sequences of coupled tasks and also fed back the design order that has the shortest design time.

6.1 Contributions and Major Findings

This thesis contributes to the existing work on engineering design management in a number of ways.

First, the literature review revealed that there is no detailed description of a process in which the DSM is applied in the product development process. This thesis contributed by providing a detailed description of the application of DSM in a practical product development process. Three design models were to define the methodology in detail using the following steps: 1) decomposing a design process; 2) building, reordering and tearing, and interpreting a DSM, and 3) numerically evaluating of the DSM. The

detailed steps represent good examples on the theory and functions behind the DSM, as well as how to use the tool.

The dimensional decision-making process of a part reveals that DSM can not only be used in the organization at a macro level, but can also help to decide micro items such as the dimensions and materials in the design process. The design of the motorized wheel focuses on how to use the DSM to organize a large design process with many tasks. In the design of the brake system, the DSM is used to create a new structure based on known information. The three models give examples of different application in the design process.

Guidelines on how to create a compact and manageable DSM are also presented in this thesis. To the author's knowledge, no such guidelines have been given in the existing literature with respect to the DSM methodology.

The calculation of the shortest design time and the contribution to the convergence of the design process corroborates the findings of Smith (1993) and Smith and Eppinger (1997). It is possible to calculate the three examples to give the design process a numerical evaluation and feedback the calculation to update their organizational structure. For example, the calculation result of the shortest design time can be used to make a new execution plan for the design of the brake system.

This thesis has shown, through an application, which the DSM is a very useful management tool in the product design process. DSM can be extended for organizing processes in manufacturing, logistics, or other fields.

6.2 Limitations of DSM

Compared to other methods, DSM is good at modeling complex design processes involving coupled design tasks. However, as a new management tool, DSM still has some shortcomings.

First, the construction process of DSM depends highly on the knowledge and experience of engineering managers and designers. For example, the relations among tasks are decided upon by people. Even though they are made based on all the opinions of team members, mistakes still cannot be avoided. Moreover, the dependency strength used in the numerical DSM is also predicted by people. Therefore, there may be some degree of error involved due to the qualitative and possibly biased inputs of individuals if enough people are not solicited for input.

Second, the DSM method introduced in this thesis is a static model, which means that relations among tasks and dependency strength are assumed to be constant in the whole design process. However, in practice, the effect of time can be significant and should be considered since, in the design process, in the light of receiving new information, information dependencies and relations and strengths may change, causing the DSM to change as well. The resulting new DSM and the subsequent analyses might suggest a different sequence of tasks, which in turn might suggest a different allocation of resources. However, the static DSM provides a good initial estimate at ordering a design process.

Third, the DSM method presented in this thesis does not provide an optimal solution, however, as previously mentioned, it does provide a good solution. In the design of product development processes, a good solution is a good guide.

Finally, the DSM tearing technology also needs to be improved. The longest circuit method, used in PSM32, is a good way of tearing DSM because the minimal assumptions are made. However, this method is qualitative because it regards all interrelations as the same strength. Therefore, it ignores the difference of the strength of interdependence.

Briefly, the limitations of DSM show that the results generated by DSM method are rough solutions for a product development process. However, even though it is not optimal, it is still significant and highly meaningful in guiding process design and implementation.

The DSM is good at modeling complex design processes involving coupled design tasks. For small design projects or those that only include sequential and parallel tasks, the advantages of DSM are not very obvious.

6.3 Future Work

The application of DSM in the design process of the motorized wheel has introduced the powerful functions of the tool in product design. Since one of the applications of the motorized wheel is to design the electrical wheelchair, the DSM methodology will be also used in the organization of the design process of an electrical wheelchair. As shown in Figure 5-3, an electrical wheelchair includes five sub-assemblies: chassis, suspension system, control system, seat, and accessories. Designers can use the spatial relationship of parts in each section to build a DSM. With the help of the DSM, designers can determine all information carefully to avoid losing necessary relations between any two parts or assemblies.

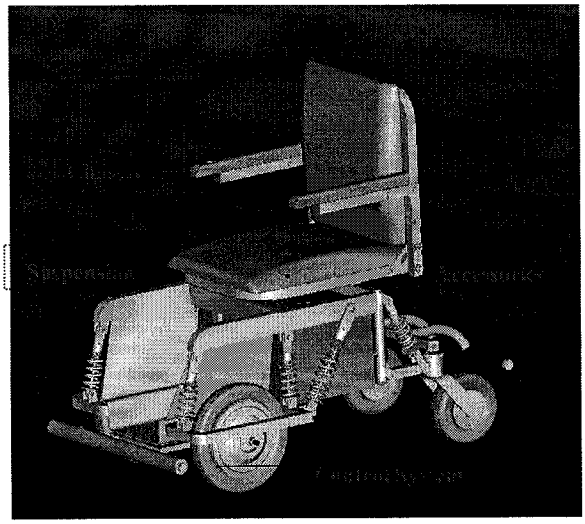


Figure 6- 1 The Motorized Wheel is applied in the Design of the Electrical Wheelchair

The design process of the electric wheelchair includes hundreds of parts, and it involves the calculation of dimensions, the choice of material, the arrangement of the control system, and so on. Using DSM, the task division and execution plan can be carefully created. For a larger design project such as a wheelchair, the advantages compared to other management tools of DSM will emerge.

BIBLIOGRAPHY

Ahmadi, R.H. and Wang, H., 1994, *Rationalizing product design development processes*, UCLA Anderson School of Management, Working paper.

Baldwin, C. Y. and Clark, K. B. 2000, *Design Rules, The Power of Modularity*, MIT Press, Volume 1. Cambridge, Massachusetts, London, England.

Bhuiyan, N., Gerwin, D., Thomson, V., 2004, *Simulation of the New Product Development Process for Performance Improvement*, Management Science, Vol. 50 No. 12, pp. 1-14.

Black, T. A., Fine, C. H., and Sachs, E. M., 1990, *A Method for Systems Design Using Precedence Relationships: An Application to Automotive Brake Systems*. M.I.T. Sloan School of Management, Cambridge, MA, Working Paper no. 3208.

Browning, T. R. and Eppinger, S. D., *A Model for Development Project Cost and Schedule Planning*, M.I.T. Sloan School of Management, Cambridge, MA, Working Paper no. 4050, November 1998.

Browning, T. R. 2001, *Applying the Design Structure Matrix to System Decomposition and Integration Problems: A Review and New Directions.* Engineering Management, Vol. 48, No. 3, pp292-305.

Clark, K.B. and Fujimoto, T., *Product Development Performance*, Harvard Business School Press, Boston, Massachusetts, 1991.

Crawford, M., 1994, *New Product Management*, 4th Edition, Irwin Co, Burr Ridge Ill., 1994.

Dorf, R. C. and Raton, E. B., 1999, *The design structure matrix in Technology Management Handbook*. Chapman & Hall /CRCnet-BASE, pp103-111.

Galbraith, J., *Designing Complex Organizations*, Addison-Wesley, Massachusetts, 1973.

Eppinger, S. D. 2001, *Innovation at the Speed of Information*, Harvard Business Review, V79, N1, pp149-158.

Ha, A.Y and Porteus, E.L., *Optimal Timing of Reviews in Concurrent Design for Manufacturability*, Management Science, Vol. 41, No. 9, September 1995, pp. 1431-1447.

Hyman, B., 2003, *Fundamentals of Engineering Design*, Second Edition, Press Education Inc., New Jersey, USA.

Kong, L. 2003, *Determining the Value of Processes and Information Structures for New Product Development*, Ph.D. Thesis, Mechanical and Industrial Engineering, McGill University, Montreal, Canada.

Krishnan, V. and Ulrich, K. T. 2001, *Product Development of Decisions: A Review of the Literature*, Management Science, V47, N1, pp1-21.

Krishnan, V., Eppinger, S., Whitney, D., 1997a, *A Model-Based Framework to Overlap Product Development Activities*. Management Science, V43, N4, pp437-451.

Krishnan, V., Eppinger, S., Whitney, D., 1997b, *Simplifying iterations in Cross-functional Design Decision Making*, J. of Mechanical Design, V119, pp485-493.

Krishnan, V. 1998, *Modeling ordered decision making in product development*, European J. of Operation Research, V111, pp351-368.

Loch. C. and Terwiesch, C., *Communication and Uncertainty in Concurrent Engineering*, Management Science, Vol. 44, No. 8, August 1998, pp. 1032-1047.

McCord, K. R. and Eppinger, S. D., *Managing the Integration Problem in Concurrent Engineering*, MIT Sloan School of Management Working Paper, no. 3594, August 1993.

Pimmler, T. U. and Eppinger, S. D., *Integration Analysis of Product Decompositions*, Proceedings of the ASME Sixth International Conference on Design Theory and Methodology, Minneapolis, MN, Sept., 1994. Also, M.I.T. Sloan School of Management, Cambridge, MA, Working Paper no. 3690-94-MS, May 1994.

Smith, R. P. 1993, *Development and Verification of Engineering Design Iteration Models*, Ph. D. Thesis, MIT Sloan School of Management, Cambridge, MA

Smith, R. P. and Eppinger, S. D. 1997a, *Identifying Controlling Features of Engineering Design iterations*, Management Science, V43, N3, pp276-293.

Smith R. P. and Eppinger, S. D. 1997b, *A Predictive Model of Sequential Iteration in Engineering Design*, Management Science, V43, N8, pp1104-1120.

Steward, Donald V., *The Design Structure System: A Method for Managing the Design of Complex Systems*, IEEE Transactions on Engineering Management, vol. 28, pp. 71-74, 1981.

Warfield, J. 1973, *Binary matrices in system modeling*, IEEE Transactions on Systems, Man, and Cybernetics, 5, pp. 441-449.

Whitney, D. E., Dong, Q., Judson, J., and Mascoli, G., *Introducing Knowledge-Based Engineering Into an Interconnected Product Development Process*, M.I.T. Center for Innovation in Product Development, Cambridge, MA, White Paper Jan. 27, 1999.

Ulrich K. T. and Eppinger S. D. 2004, *Product Design and Development*, McGraw Hill.

Ulrich K., *Introduction to the Special Issue on Design and Development*, Management Science, January 2001, Vol.47, No.1.

Yassine, A. *et al*, *A Decision Analytic Framework for Evaluating Concurrent Engineering*, IEEE Transactions on Engineering Management, Vol. 46, No. 2, May 1999, pp. 144-157.

APPENDIX A

PSM32 - A Professional Software for Partitioning and Tearing DSM

PSM 32, created by Problematics Inc., is a software that specializes in the treatment of DSM. This software is convenient for partitioning and tearing DSM. The details of setting-up and using this software can refer to the tutorial in the website www.problematics.com.

The software mainly supplies us three applications for analyzing DSM:

- Partition DSM based on the rule that is introduced in section 3.2.3. Especially for some big matrices, this software is more efficient to arrange rows and columns than human brain.
- Tear DSM to break the principal circuit. For example, the longest information circuit of the block in Figure A-1 is the circle from task 2, to task 1, and to task 3, shown as Figure A-1(a). If we choose task 3 as tear, all circuits will be broken, shown as A-1(b). In this process, we make an assumption to take task 3 out, so the feedbacks from task 3 are known to the other tasks temporarily, and we use number 5 to express them. For a big block, if we cannot tear the whole block in one instance, we would have to take another tear. The final torn result is like Figure A-1(c), and all 0s are moved to the left-diagonal area.
- Generate the implementation plan. After the DSM is totally torn, the software gives out precedence of execution of tasks to help sequence tasks.

	1	2	3
1	*	[illegible]	0+
2	0+	*	[illegible]
3	[illegible]	0+	*

Figure A- 1(a) The Principle Circuit (marked by 0+)

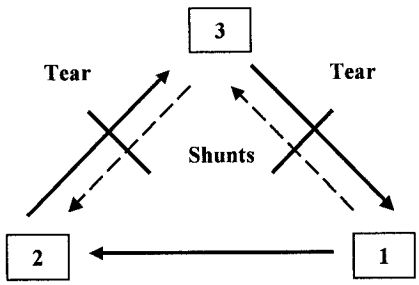


Figure A- 2 (b) Principle Circuit and Shunts

Deleted: 1

	1	2	3
1	*		5
2	[illegible]	*	5
3	[illegible]	[illegible]	*

Figure A- 3 (c) The Torn Block

Deleted: 1

APPENDIX B

Sensitivity Analysis on DSM Rework Probabilities

Recall from Chapter 5 that the original work transformation matrix is:

	A	B	C	D
A	3	0	0.1	0
B	0.1	7	0.3	0
C	0.1	0.2	5	0.1
D	0	0.1	0.1	2

Figure 5- 5 Probabilities of Rework and Task Execution Time

Deleted: 2

The execution time of the sequence ABCD was calculated as:

$$f1 = r_D + s_C + t_B + u_A = 2.845 + 8.050 + 7 + 3 = 20.895 \quad \text{(B-1)}$$

Now in order to check the sensitivity of the dependency strengths, a_{21} is changed from 0.1 to 0.2, which means that the dependency strength that task B depends on task A is increased.

Calculation of the Minimal Execution Time

Compared to the Markov chain in Figure 5-3, only the rework probability between A and B is changed, shown in Figure B-1.

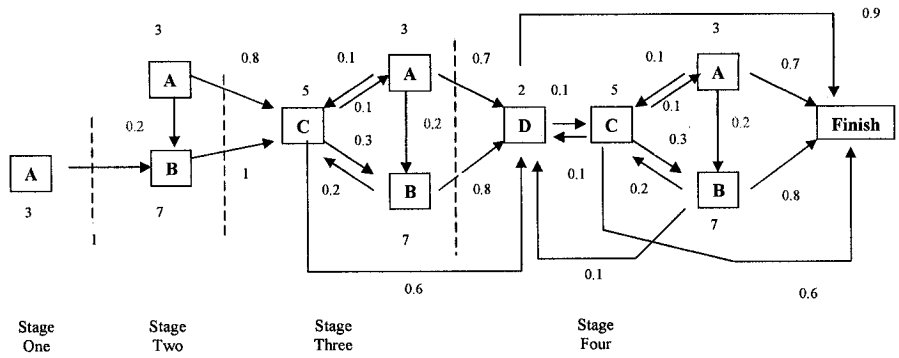


Figure B- 1 Markov Chain of Sequence ABCD (changing one Rework Probability)

In the 4th stage,

$$\begin{bmatrix} 1 & -0.2 & -0.1 & 0 \\ 0 & 1 & -0.2 & -0.1 \\ -0.1 & -0.3 & 1 & -0.1 \\ 0 & 0 & -0.1 & 1 \end{bmatrix} \begin{bmatrix} r_A \\ r_B \\ r_C \\ r_D \end{bmatrix} = \begin{bmatrix} 3 \\ 7 \\ 5 \\ 2 \end{bmatrix}$$

$r_D = 2.856$

In the 3rd stage,

$$\begin{bmatrix} 1 & -0.2 & -0.1 \\ 0 & 1 & -0.2 \\ -0.1 & -0.3 & 1 \end{bmatrix} \begin{bmatrix} s_A \\ s_B \\ s_C \end{bmatrix} = \begin{bmatrix} 3 \\ 7 \\ 5 \end{bmatrix}$$

$s_C = 8.143$

In the 2nd stage,

$t_A = 0.2 t_B + 3$

$t_B = 7$

In the 1st stage,

$u_A = 3$

The total execution time:

$$r_D + s_C + t_B + u_A = 2.856 + 8.143 + 7 + 3 = 20.999 \quad \text{(B-2)}$$

Comparison of the results between (B-1) and (B-2) yields:

Table B- 1 Comparison of Execution Time-Changing one Rework Probability (Sequence ABCD)

Comparison Items	Original	Current	Change
Dependency, a_{21}	0.1	0.2	+100%
4th Stage, r_D	2.845	2.856	+0.38%
3rd Stage, s_C	8.050	8.143	+1.15%
2nd Stage, t_B	7	7	0
1st Stage, u_A	3	3	0
Total	20.895	20.999	**+0.49%**

Now check the sequence BACD:

Table B- 2 Comparison of Execution Time-Changing one Rework Probability (Sequence BACD)

Comparison Items	Original	Current	Change
Dependency, a_{21}	0.1	0.2	+100%
4th Stage, r_D	2.845	2.856	+0.38%
3rd Stage, s_C	8.050	8.143	+1.15%
2nd Stage, t_A	3.7	4.4	+18.92
1st Stage, u_B	7	7	0
Total	21.595	22.399	**+3.72%**

Thus, if one element in the work transformation matrix is changed, the influence varies with different execution sequences. In this example, the changed element is relative to the rework that the task A causes to the task B. Thus, for the sequence that this element (dependency strength) does not work on, the difference is minimal, but for the sequence that it works on, there is a difference, though it is not significant.

Calculation of the Work Vectors

The original dependency strength matrix,

$$A = \begin{bmatrix} 0 & 0 & 0.1 & 0 \\ 0.1 & 0 & 0.3 & 0 \\ 0.1 & 0.2 & 0 & 0.1 \\ 0 & 0.1 & 0.1 & 0 \end{bmatrix}$$

The work vectors are calculated based on this matrix:

$$u_0 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad u_1 = \begin{bmatrix} 0.1 \\ 0.4 \\ 0.4 \\ 0.2 \end{bmatrix} \quad u_2 = \begin{bmatrix} 0.04 \\ 0.13 \\ 0.11 \\ 0.08 \end{bmatrix} \quad u_3 = \begin{bmatrix} 0.011 \\ 0.037 \\ 0.038 \\ 0.024 \end{bmatrix} \qquad \text{(B-3)}$$

Now change the dependency strength matrix to A1, with the same change as above:

$$A1 = \begin{bmatrix} 0 & 0 & 0.1 & 0 \\ 0.2 & 0 & 0.3 & 0 \\ 0.1 & 0.2 & 0 & 0.1 \\ 0 & 0.1 & 0.1 & 0 \end{bmatrix}$$

The work vectors in the first four stages are:

$$u_0 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad u_1 = \begin{bmatrix} 0.1 \\ 0.5 \\ 0.4 \\ 0.2 \end{bmatrix} \quad u_2 = \begin{bmatrix} 0.04 \\ 0.14 \\ 0.13 \\ 0.09 \end{bmatrix} \quad u_3 = \begin{bmatrix} 0.013 \\ 0.047 \\ 0.041 \\ 0.027 \end{bmatrix} \qquad \text{(B-4)}$$

Comparison of work vectors between (B-3) and (B-4) is as follows:

Table B- 3 Comparison of Work Vectors-changing one Rework Probability

	Initial (1st Stage)	2nd Stage				3rd Stage				4th Stage			
		A	B	C	D	A	B	C	D	A	B	C	D
Original	-	0.1	0.4	0.4	0.2	0.04	0.13	0.11	0.08	0.011	0.037	0.038	0.024
Current	-	0.1	0.5	0.4	0.2	0.04	0.14	0.13	0.09	0.013	0.047	0.041	0.027
Changed	-	0	**0.1**	0	0	0	0.01	0.02	0.01	0.002	0.01	0.003	0.003

The amount of total work vectors has been changed as follows:

Table B- 4 Difference of total Work Vectors-changing one Rework Probability

Tasks	Amount
A	+0.002
B	+0.12
C	+0.023
D	+0.013

With the dependency strength a_{21} changed from 0.1 to 0.2, the total rework on task B is increased by 12%, but for other tasks, the rework increase is minimal, for example, task A increases by 0.2%, task C by 2.3%, and task D by 1.3%.

From the results in (B-4), the sequence that takes the longest design time is still BCDA, and the sequence of the minimal execution time is still ADCB (no change from original results with the unmodified DSM).

Now, if two rework probabilities are changed in the original work transformation matrix simultaneously, i.e., b_{23} is changed from 0.3 to 0.4, and c_{32} is changed from 0.2 to 0.1, the modified matrix is:

	A	B	C	D
A	3	0	0.1	0
B	0.1	7	**0.4**	0
C	0.1	**0.1**	5	0.1
D	0	0.1	0.1	2

Figure B- 2 A new Work Transformation Matrix-changing two Rework Probabilities

Calculation of the Minimal Execution Time

For the execution order ABCD

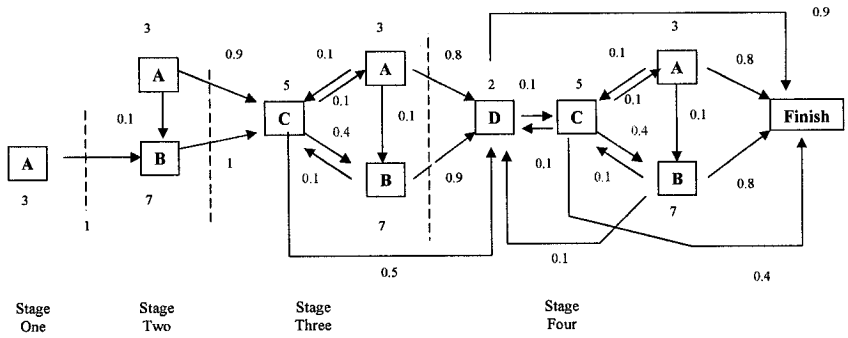


Figure B- 3 Markov Chain of Sequence ABCD (changing two Rework Probabilities)

In the 4th stage,

$$\begin{bmatrix} 1 & -0.1 & -0.1 & 0 \\ 0 & 1 & -0.1 & -0.1 \\ -0.1 & -0.4 & 1 & -0.1 \\ 0 & 0 & -0.1 & 1 \end{bmatrix} \begin{bmatrix} r_A \\ r_B \\ r_C \\ r_D \end{bmatrix} = \begin{bmatrix} 3 \\ 7 \\ 5 \\ 2 \end{bmatrix}$$

$r_D = 2.904$

In the 3rd stage,

$$\begin{bmatrix} 1 & -0.1 & -0.1 \\ 0 & 1 & -0.1 \\ -0.1 & -0.4 & 1 \end{bmatrix} \begin{bmatrix} s_A \\ s_B \\ s_C \end{bmatrix} = \begin{bmatrix} 3 \\ 7 \\ 5 \end{bmatrix}$$

$s_C = 6.502$

In the 2nd stage,

$t_A = 0.2 t_B + 3$

$t_B = 7$

In the 1st stage,

$u_A = 3$

The total execution time:

$$r_D + s_C + t_B + u_A = 2.904 + 6.502 + 7 + 3 = 19.406 \quad \text{(B-5)}$$

For the execution sequence DCBA:

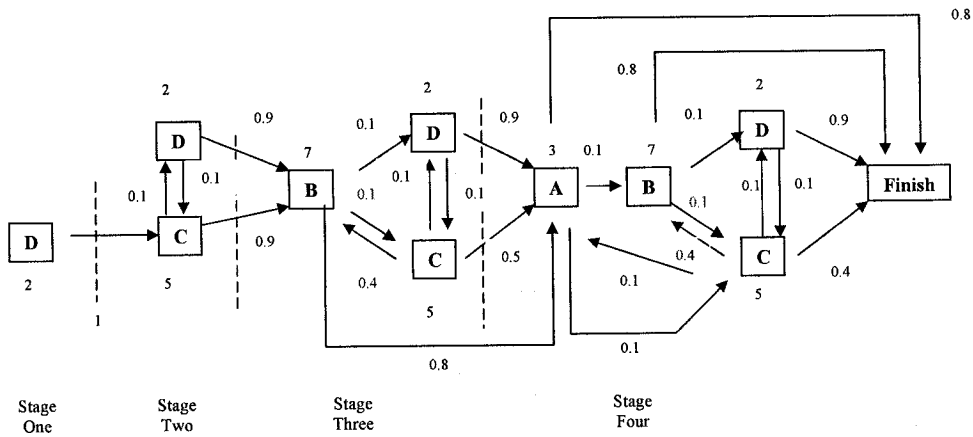


Figure B- 4 Markov Chain of Sequence DCBA (changing two Rework Probabilities)

In the 4th stage,

$$\begin{bmatrix} 1 & -0.1 & 0 & 0 \\ -0.1 & 1 & -0.4 & -0.1 \\ -0.1 & -0.1 & 1 & 0 \\ 0 & -0.1 & -0.1 & 1 \end{bmatrix} \begin{bmatrix} r_D \\ r_C \\ r_B \\ r_A \end{bmatrix} = \begin{bmatrix} 2 \\ 5 \\ 7 \\ 3 \end{bmatrix}$$

$r_A = 4.723$

In the 3rd stage,

$$\begin{bmatrix} 1 & -0.1 & 0 \\ -0.1 & 1 & -0.4 \\ -0.1 & -0.1 & 1 \end{bmatrix} \begin{bmatrix} s_D \\ s_C \\ s_B \end{bmatrix} = \begin{bmatrix} 2 \\ 5 \\ 7 \end{bmatrix}$$

$s_B = 8.140$

In the 2nd stage,

t_C =5.253

In the 1st stage,

$u_D = 2$

The total execution time: $r_A + s_B + t_C + u_D$ =4.723+8.140+5.253+2=20.116 (B-6)

Again, a comparison of the results yields:

Table B- 5 Comparison of Execution Time-Changing two Rework Probabilities (Sequence ABCD and DCBA)

Comparison Items	Original	Current	Change
Dependency, a_{23}	0.3	0.4	+33%
Dependency, a_{32}	0.2	0.1	-50%
Total Time of ABCD	20.895	19.406	**-7.1%**
Total Time of DCBA	20.867	20.116	**-3.6**

Changing two rework probabilities simultaneously, the influence on the execution time is somewhat noticeable. It is important to verify if there is a change in recommended sequence of tasks through the calculation of the work vectors.

Calculation of the total Vectors

$$A2 = \begin{bmatrix} 0 & 0 & 0.1 & 0 \\ 0.1 & 0 & 0.4 & 0 \\ 0.1 & 0.1 & 0 & 0.1 \\ 0 & 0.1 & 0.1 & 0 \end{bmatrix}$$

Calculate the work vectors:

$$u_0 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad u_1 = \begin{bmatrix} 0.1 \\ 0.5 \\ 0.3 \\ 0.2 \end{bmatrix} \quad u_2 = \begin{bmatrix} 0.03 \\ 0.13 \\ 0.08 \\ 0.08 \end{bmatrix} \quad u_3 = \begin{bmatrix} 0.008 \\ 0.035 \\ 0.024 \\ 0.021 \end{bmatrix} \quad \text{(Modified)}$$

$$u_0 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad u_1 = \begin{bmatrix} 0.1 \\ 0.4 \\ 0.4 \\ 0.2 \end{bmatrix} \quad u_2 = \begin{bmatrix} 0.04 \\ 0.13 \\ 0.11 \\ 0.08 \end{bmatrix} \quad u_3 = \begin{bmatrix} 0.011 \\ 0.037 \\ 0.038 \\ 0.024 \end{bmatrix}$$ (Original)

Comparing the modified results with the original results, it can be seen that there is only a small difference. The sequences of the longest and shortest execution time do not change and are still BCDA and ADCB, respectively.