Available online at www.sciencedirect.com
SCIENCE DIRECT

Expert Systems with Applications 31 (2006) 705–714

Expert Systems with Applications

www.elsevier.com/locate/eswa

Structure representation for concurrent analysis of product assembly and disassembly

Jyh-Cheng Yu* , Yi-Ming Li

Department of Mechanical and Automation Engineering, National Kaohsiung First University of Science and Technology
No. 2 Juoyue Rd., Kaohsiung 811, TAIWAN, R.O.C.

The original publication is available at http://dx.doi.org/10.1016/j.eswa.2006.01.039.

Abstract

This paper presents a simple and novel structure representation supporting the assembly and disassembly planning of electromechanical products. The proposed Relationship Matrix derived from a directed graph represents both the information of the component connectivity and the layout precedence of functional elements. The feasible assembly and disassembly sequences and the minimum service steps for the malfunction component can be easily derived with the corresponding inference rules. The structure representation and the inference kernel can be readily applied to future concurrent design review for assemblability, serviceability, and recyclability. A prototype software tool is introduced to demonstrate the application of the proposed scheme.

Keyword: Layout Design, Concurrent Design, Service Mode Analysis, Relationship Matrix, Assembly Sequence Planning

1. Introduction

Product layout designs describe the arrangement of composing components in the product space. Design engineers have to contemplate various life cycle issues to determine the best orientation, assembling methods, and module designs of the components. Layout designs will affect assembly and disassembly sequences that are crucial factors in deign for assembly[1], serviceability[8], and recyclability[2]. Design for assembly advocates to improve assembly process by reducing part counts, making individual parts easier to assemble, and reducing the possibility of assembly errors. Assembly sequencing is important to assess the design efficiency and the search of optimal assembly planning. Design for disassembly [12], on the other hand, addresses maintainability and recyclability, such that flawed components can be easily accessed and replaced in product maintenance, and valuable resources can be efficiently retrieved at the end of the product life cycle. Here, incomplete disassembly is often preferred to dismantle only the components blocking the access to the serviced part. However, the analysis of assembly and disassembly should be a continuous trade-off of life cycle factor in a simultaneous fashion. Optimum product architecture has to be determined to reduce the life cycle cost.

To design an expert system for concurrent engineering, the automatic sequencing of assembly and disassembly is essential. Conventional CAD models represent only orientation and geometric information of product assemblies. However, the assembly methods and the spatial interaction among composing components are not manifest. Several modeling strategies have been proposed for the design representation of product architecture, such as directed graph [10], the liaison diagram [4], and the Component-Fastener Graph [9], to record the connectivity and the interference information. Homem and Sanderson [6] proposed the AND/OR graph, a compact representation of all possible assembly plans of a given product. Some suggested the directional Interference Matrices from a CAD model to describe the spatial

* Corresponding author. Tel:+886-7-6011000 ext.2228; Fax: +886-7-6011066 (J.-C. Yu)
E-mail: jcyu@ccms.nkfust.edu.tw

interference in disassembly procedure and to automatically generate all the possible assembly[5] and disassembly[7] sequences. However, detailed CAD models are often not available at the early stage of design, and slanted insertion of components can't be described using the interference matrices of principal directions.

This paper presents a simple and novel structure representation for conceptual design to support assembly and disassembly planning of fixed-form products. The structure representation and the proposed inference rules can derive feasible assembly sequences and the minimum disassembly sequence for any specified part. The scheme can be readily applied to the concurrent analysis for conceptual designs. A prototype system is then described to demonstrate the application.

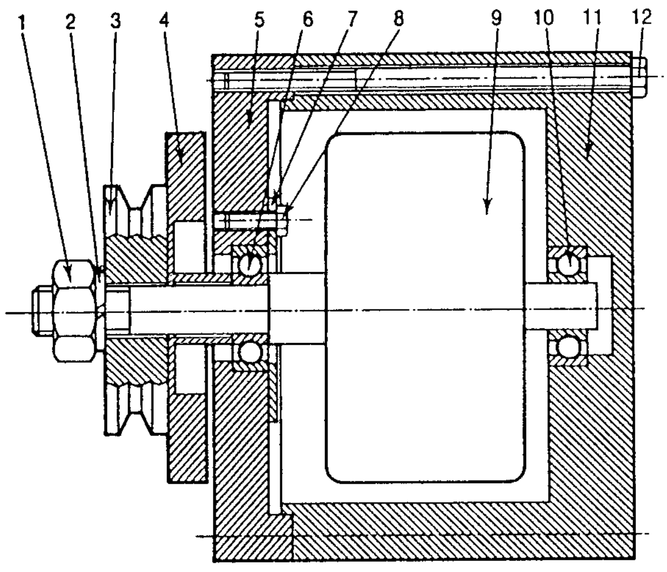
2. Product Assembly and Disassembly

Ease of assembly and disassembly is the common requirement in product life-cycle designs such as design for assembly, serviceability, and recyclability. Product assembly can be deemed as a process to establish the structure relationship among composing components. The process often starts from a base part, such as the Rear Housing in the example of Figure 1, secured in a fixture, and adds the following components to the main assembly. Auxiliary tools, such as screwdrivers, wrenches, etc., might be required to complete the assembly. The sequencing should consider the feasibility and ease of assembly. For instance, the components assembled earlier should not obstruct the assembly of subsequent components. Also, we should group the assembly steps using the same tools to reduce the change of workstation.

To ease the assembly effort, some components might be grouped together to form a subassembly before attaching to the main assembly. The subassemblies could be optional or mandatory. The constituents forming the optional subassemblies can also be attached to the main assembly sequentially instead of a group. On the contrary, the constituents of the mandatory subassembly have to be assembled beforehand; otherwise they cannot be inserted to the main assembly. For the example in Figure 1, the Front Bearing must be inserted to the Front Housing, and fastened by the Retainer and three Short Screws to compose a Front Module before assembled to the Rear Housing. This Front Module is a mandatory subassembly because the constituents cannot be sequentially put into the main assembly based on the Rear Housing.

Product service is inevitable during the life cycle. If certain parts fail, technicians need to retrieve them for maintenance or replacement. The objects blocking the access and those connected to the flawed part have to be removed before the service. To reduce the labor cost, those requiring frequent service should be placed in the outer region of structure layout and adopt the assembly methods that can sustain repeat reassembly. The failure frequencies and the maintenance procedures of possible service modes have to be determined at the early stage of design to assess the warranty cost.

The structure layout design should consider all the life cycle issues such as manufacturing, maintenance, and recycling. Due to the lack of design representation of product layout, the analysis of product assembly and disassembly usually occur late at the prototyping stage, which often incurs higher design cost if any modification is required. The following sessions will describe the proposed structure representation for fixed-form constituents and the inference rules for assembly and disassembly sequences, which can be applied to the expert system for layout design.



	Part Name
1	Nut
2	Lock Washer
3	Pulley
4	Fan
5	Front Housing
6	Front Bearing
7	Retainer
8	Short Screw × 3
9	Rotor
10	Rear Bearing
11	Rear Housing
12	Through Screws × 3

Figure 1. The product layout of an automotive generator

3. Representation of Product Structure

3.1. The Constituent Objects of the Product

Product Function Analysis [3] is often used in the conceptual design to transform customer requirements to product functions. Design engineers then conceive specific objects and mechanism to realize the terminal functions. The assembly of the conceived objects composes a conceptual design. To obtain the product stability and rigidity, connecting features and fasteners, such as screw, bolts, and retaining rings, will be introduced.

To reduce the searching space of assembly and disassembly sequences, this study classifies the product constituents into the functional elements and the fasteners (Figure 2). The functional elements are the constituents that realize certain customer requirements. The fasteners, such as screws, rivets, retaining rings, etc, on the other hand, are auxiliary objects mainly to maintain structure stability and rigidity. Because the fasteners always accompany the corresponding parts to fasten during the assembly procedure, once the correct assembly order of the functional elements is determined, the complete assembly steps can be readily derived.

For the example of the automotive generator shown in Figure 1, the classification of the composing objects is presented in Table 1. Another reason to identify the fasteners in a product structure is that the fasteners are usually the targets to redesign for assembly. DFA encourages incorporating the fastening feature to the functional element to reduce the part count. Designers should only have legitimate reasons to use auxiliary fasteners. Also, the fastening method should consider the ease of disassembly and reassembly if the functional element to secure will need service during the life cycle.

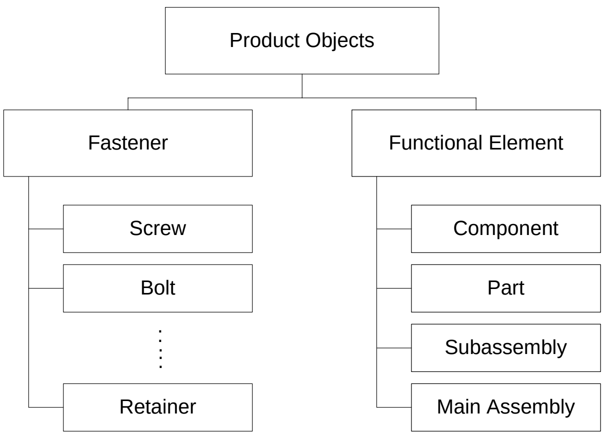


Figure 2. Classifications of Product Objects

Table 1.

The classification of the composing object of the generator

Functional elements	Fasteners
Pulley	
Fan	Nut
Front Housing	Lock Washer
Front Bearing	Short Screws (3)
Rotor	Through Screws (3)
Rear Bearing	Retainer
Rear Housing	

3.2. The Object Relationships

We divide the relationships among functional elements into two categories: the Physical Link and the Layout Interference. The Physical Link represents direct connection of two objects by either geometrical interlock or auxiliary fasteners. The other relationship that is termed the “Layout Interference” exists when an object affects the assembly sequence of other objects due to spatial arrangement, even though there is no direct contact among these objects. It is impossible to finish the assembly if certain object is assembled before the other when there is a Layout Interference between these two objects. Here the interference relationship is a general concept and doesn’t differentiate the interfering direction because the detailed object geometry is often undecided at the conceptual design stage.

For instance in Figure 3, the relationship between objects *C* and *B* is a Physical Link since object *C* is connected to object *B* by a snap feature. Another example of Physical Link exists between objects *D* and *B*; however, the connectivity is established by a screw fastener. On the other hand, there is no direct contact between objects *F* and *E*, but object *E* must precede object *F* in the assembly sequence due to the covering effect. This remote relationship between objects *F* and *E* is termed the “Layout Interference”.

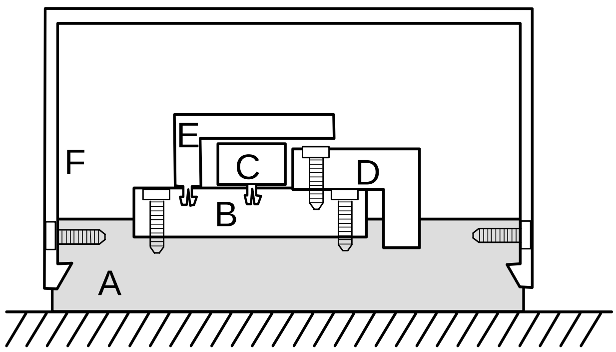


Figure 3. Example of product layout

This study combines the concept of interference and connectivity, and proposes the Object-Relationship Graph (OR-Graph) to represent the product structure. The attributes of the object contain information of the object category, the part name, the material type, and the part weight, while the attributes of the relationship contain the relationship type, the assembly method, and the fastener type if individual fasteners are applied. During the assembly, one object usually has a precedence preference over the other because of size, weight, and ease of assembly. Although it is not necessary for all the Physical Links, the pre-assigned information can greatly reduce the searching space of assembly and disassembly sequences.

The OR-Graph is a directed graph where the arrow direction indicates the next assembly occurrence. For each relationship O_i-O_j, O_i is called the *related object* and O_j is called the *relating object* if O_j precedes O_i in the assembly sequence because of either the assembly conventions or the Layout Interference. The OR-Graph is established from the *relating object* to the *related object*. The corresponding OR-Graph for the previous layout is shown in Figure 4. For examples, the direction of the dashed link connecting objects C and E indicates a Layout Interference that C should precede E in the assembly sequence otherwise object C cannot be installed. The direction of the solid link connecting objects B and A indicates a Physical Link that A precedes B due to ease of assembly.

3.3. Relationship Matrix

The OR-Graph can be described as a matrix form, the Relationship Matrix (*RM*), to facilitate the derivation of the assembly and the disassembly sequences. The Relationship Matrix is an $n \times n$ matrix recording the object relationships where n is the number of the functional elements. The functional elements are listed in the columns and the rows of *RM* in the same order. The rows indicate the *related objects*, and the columns indicate the

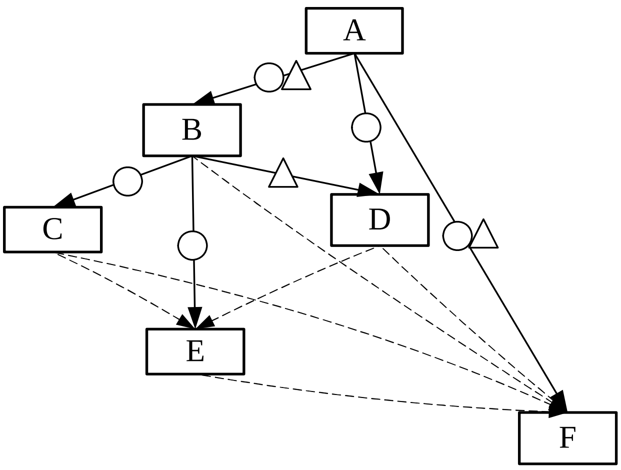


Figure 4. The OR-Graph of Figure 3

Table 2.

The Relationship Matrix of the example layout in Figure 3

R-ed

R-ing

$$RM = \begin{array}{c c} & \begin{array}{cccccc} F & D & C & B & E & A \end{array} \\ \begin{array}{c} F \\ D \\ C \\ B \\ E \\ A \end{array} & \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 2 \\ 0 & 0 & 0 & 3 & 0 & 2 \\ 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 \\ 0 & 1 & 1 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

relating objects. The relationship between O_i and O_j is denoted in $RM(i, j)$. The corresponding Relationship Matrix of the layout example in Figure 3 is shown in Table 2. The notations used in this paper are explained as follows:

(a) $RM(i, j)$=0 if there is no relationship between O_i and O_j.

(b) $RM(i, j)$=1 if there is a Layout Interference between two objects O_i and O_j. For instance in Figure 3, object *E* will prevent the assembly of *C* and *D* if *E* is installed before them. Here *C* is the *related object* for this interference relationship, and the entries at row *E* and columns *C* and *D* are denoted by “1”.

(c) $RM(i, j)$=2 if there is a Physical Link between O_i and O_j. O_i is the *related object* and O_j is the relating counterpart of the relationship pair. For instance in Figure 3, object *E* is connected to object *B* with a snap feature. *E* is the *related object* and *B* is the *relating object*. The entry at row *E* and column *B* is denoted by “2”.

(d) $RM(i, j)$=3 if there are both a Physical Link and a Layout Interference between O_i and O_j, such as object *D* and object *B* in Figure 3.

The Relationship matrix of a more realistic example such as the automotive generator of Figure 1 is shown in Table 3. The matrix considers only the functional elements to reduce the dimension. The object order in the initial *RM* is randomly selected. The *relating objects* are identified for each relationship of *related object*, and their corresponding relationship codes are filled into the entries. For example, the Pulley is inserted to the Rotor and in contact with the Fan. Also, the Pulley will interfere the assembly of the Fan if installed first. Therefore, the entries of the row “Pulley” at the columns “Fan” and “Rotor” are 3 and 2 respectively. Two fasteners, the Nut and the Lock Washer, secure the relationship between the Pulley and the Rotor, which are not shown in the matrix and will be introduced later when the complete assembly steps of the Pulley is considered.

Table 3

The Relationship Matrix of the automotive generator

	Pulley	Fan	Front housing	Front bearing	Rotor	Rear bearing	Rear housing
Pulley		3	0	0	2	0	0
Fan	0		0	0	2	0	0
Front housing	0	0		1	1	1	2
Front bearing	0	0	2		2	0	0
Rotor	0	0	0	0		3	0
Rear bearing	0	0	0	0	0		2
Rear housing	0	0	0	0	0	0	

4. The Inference of Assembly Sequences

4.1. The Characteristics of Rational Relational Matrix

When the row order of the objects from the top of the matrix represents a feasible assembly sequence, the Relationship Matrix is termed rational and will present a particular pattern. The matrix is often a lower triangular matrix with all zero entries above the diagonal. For instance, Table 4 is a rationalized *RM* of the example layout in Figure 3.

(1) No Physical Link exists in the entries located at the upper triangular of the rational Relationship Matrix, RM_r. Because the rows represent the *related objects* of the relationship pairs, if an object will only assemble to the precedent objects, there will be no Physical Link between the object and the following objects, and the row order will be a feasible assembly sequence.

(2) The number of Layout Interference in the upper triangular of the RM_r is minimized. For a feasible assembly sequence, the earlier objects should not obstruct the assembly of later objects. Therefore, there should be no Layout Interference in the upper triangular of the RM_r unless the object is the base of a mandatory subassembly that will be addressed in the later section.

Table 4

The rationalized Relationship Matrix of the example in Figure 3

$$RM_r = \begin{array}{c} \text{R-ing} \xrightarrow{\text{R-ed}} \\ \\ \end{array} \begin{array}{c|cccccc} & A & B & C & D & E & F \\ \hline A & 0 & 0 & 0 & 0 & 0 & 0 \\ B & 2 & 0 & 0 & 0 & 0 & 0 \\ C & 0 & 2 & 0 & 0 & 0 & 0 \\ D & 2 & 3 & 0 & 0 & 0 & 0 \\ E & 0 & 2 & 1 & 1 & 0 & 0 \\ F & 2 & 1 & 1 & 1 & 1 & 0 \end{array}$$

4.2. The Inference of a Feasible Assembly Sequence

The initial construction of the Relationship Matrix doesn't guarantee rational. Through the following row and column operations, we could rationalize the *RM* and obtain a feasible assembly sequence:

(1) Move the entries with the Physical Link on the upper triangular to the lower triangular by exchanging the object order. If $RM(i, j)$ is "2" or "3" and $j>i$, interchange columns i and j and rows i and j. The purpose is to rearrange the object order in the *RM* such that for all the Physical Links the *relating objects* will always precede their *related* counterparts.

(2) Move the entries with the Layout Interference on the upper triangular to the lower triangular by interchanging the order of two objects. If $RM(i, j)$ is "1", $j>i$, and $RM(j, i) \neq 2$, interchange columns i and j and rows i and j. This procedure will assure the number of Layout Interference left in the upper triangular of the *RM* is minimized without violating the precedence requirement in the last step.

Equation (1) represents the matrix operation for the interchange of objects i and j. Repeat the transformations until that the following two conditions are met. The final Relation Matrix is called rational RM_r.

$$RM_{ij} = [E_{ij}]RM\,[E_{ij}] \tag{1}$$

$$[E_{ij}] = \begin{array}{c|c} & \begin{array}{cccccccccc} & & i & & & j & & & & \end{array} \\ \begin{array}{c} \\ \\ \\ i \\ \\ \\ j \\ \\ \\ \\ \end{array} & \begin{bmatrix} 1 & \cdots & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 1 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & \cdots & 0 & 0 & 1 & 0 & 0 & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 0 & 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 1 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 0 & 0 & 0 & 0 & 1 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 1 \end{bmatrix} \end{array} \tag{2}$$

Rational condition #1:
For $j > i$, $RM_r(i, j) \neq 2$ and $RM_r(i, j) \neq 3$.

Rational condition #2:
For $j > i$, $RM_r(i, j) \neq 1$, unless $RM_r(j, i) = 2$.

The rational conditions will be reached if the objective function of Eq. (3) is minimized. The rational transformation Ω is the product of a series interchange operations.

$$objective = \sum_{j=i+1}^{n} \sum_{i=1}^{n} RM\,(i, j) \tag{3}$$

$$RM_r = \Omega \cdot RM \cdot \Omega \tag{4}$$

The rational RM_r is not unique. We can continue to search for other rational RM_r's. However, the proposed methodology is not intended for the exhaustive search of assembly sequences. The object order listed in the column or the row of every rationalized RM_r would infer a feasible assembly sequence. For instance, Table 4 is a rationalized *RM* of the example layout in Figure 3. Therefore the object order in the RM_r, i.e., *A*, *B*, *C*, *D*, *E*, and *F*, represents a feasible assembly sequence. Because the rational matrix contains only the functional elements, the complete assembly sequence can be obtained by inserting the auxiliary fasteners after the corresponding objects to be constrained.

4.3. Recognition of Mandatory Subassemblies

If the product structure contains any mandatory subassembly, we cannot assemble all the objects to the main assembly in a sequential order. All the members in a mandatory subassembly have to be assembled in advance to form a module before inserting to the main assembly. The existence and the members of the mandatory subassembly can be easily recognized from the rationalized Relationship Matrix.

Theorem #1: If the Relationship Matrix RM_r is rational and $\exists\, RM_r(i, j) = 1$, $j > i$, the structure contains at least one mandatory subassembly. The number of the mandatory subassemblies is equal to the number of rows contains entry "1" in the upper triangular of RM_r.

Theorem #2: The sub-base of the mandatory subassembly is the object in row i where $RM_r(i, j) = 1$ and $j > i$. The members of the mandatory subassembly are the objects appeared in the upper triangular and interfered by the sub-base.

If the Relationship Matrix, $RM_r(i, j) = 1$, $j > i$, the interchange of objects i and j is prohibited because $RM_r(j, i)$ must be 2 or 3, and the interchange will violate the rational conditions. Therefore, object i cannot be attached directly to the main assembly, and must be the sub-base of a mandatory subassembly. Those interfered by and appeared after object i are the members of the mandatory subassembly, and have to be assembled first. The completed subassembly will then be assembled to the main assembly.

Table 5 is the rationalized Relationship matrix of the automotive generator in Figure 1. The matrix satisfies both the rational conditions. The entries at the upper triangular are all zero except the one of the row "Front Housing" and column "Front Bearing". The Layout

Table 5.
The rationalized Relationship Matrix, RM_r, of the automotive generator

	Rear housing	Rear bearing	Rotor	Front housing	Front bearing	Fan	Pulley
Rear housing		0	0	0	0	0	0
Rear bearing	2		0	0	0	0	0
Rotor	0	3		0	0	0	0
Front housing	2	1	1		1	0	0
Front bearing	0	0	2	2		0	0
Fan	0	0	2	0	0		0
Pulley	0	0	2	0	0	3	

Table 6.

The Sub-Relationship Matrix of Front Cover Module of the automotive generator

Sub-*RM* =

	Front housing	Front bearing
Front housing		1
Front bearing	2	

Interference could not be moved to the lower triangular because the entry at row "Front Bearing" and column "Front Housing" is "2". According to the theorem #1, there is a mandatory subassembly and the Front Housing is the sub-base. The members of the subassembly include the Front Bearing that is interfered by and appeared after the "Front Housing". During the assembly, the "Front Bearing" is fastened to the "Front Housing" by a "Retainer" and three "Short Screws" to form the "Front Cover Module". The "Front Cover Module" is then assembled to the main assembly by the Physical Links between the members of the subassembly and the rest of the functional elements. In this case, the subassembly is inserted to the "Rotor" and attached to the "Rear Housing" by three "Through Screws".

After the recognition of mandatory subassemblies, the Relationship Matrix can be divided into certain sub Relationship Matrices, Sub-*RM*, and the main Relationship Matrix, Main-RM_r. The Sub-*RM* describes the relationship among the members of the subassembly such as the Front Cover Module in the example of the automotive generator. The relationship between the subassembly and the rest of the functional elements are combined from the members of the subassembly as shown in Table 7.

4.4. Generation of Assembly Steps

The row order in the rationalized Relationship Matrix provides a feasible sequence of the structure. The fastening methods are the attributes of the entries of the *RM*. The insertion of the fasteners to secure the corresponding objects will generate the complete assembly steps. The assembly of the mandatory subassembly should be done first.

5. The Inference of Disassembly Sequences

To retrieve a malfunction part for service, technicians have to remove the objects blocking the access and disconnect all the objects attached to the target part. We can identify these objects from the rational Relationship Matrix using the depth-first-search method. The inference rules will provide the minimum disassembly and reassembly procedure, and some objects will be removed as a subassembly if further disassembly is not required. The inference scheme is as follows:

(1). The serviced object is first added to the Disassembly Queue (*DQ*) that is a First-In-Last-Out processing queue.

(2). Add the *related objects* that have the relationship of either the Physical Link or the Layout Interference to the serviced part to the *DQ*. These objects can be identified from the column of the serviced part, and are put to the *DQ* according to their assembly order.

(3). Take one object, O_t, from the *DQ*, and check the disassemblability. If the object O_t is not interfered by any other object, i.e. no entries 1 or 3 in the column O_t in the rational Relationship Matrix, remove the target object. Otherwise, add those interfering the target object to the *DQ*. If there is a object O_k such that $RM(k, t)=2$, O_t can be removed with O_k still attached to it. The disconnection between O_t and O_k is not required because O_k will not interfere the removal of O_t. The Relationship Matrix is then reduced by removing the columns and the rows that have been removed.

(4). Repeat step (3) until all the objects in the *DQ* are removed.

Table 7.

The Main-Relationship Matrix of the automotive generator

Main-RM_r =

	Rear housing	Rear bearing	Rotor	Front Cover module	Fan	Pulley
Rear housing		0	0	0	0	0
Rear bearing	2		0	0	0	0
Rotor	0	3		0	0	0
Front cover module	2	1	3		0	0
Fan	0	0	2	0		0
Pulley	0	0	2	0	3	

Consider the object A of the example layout in Figure 3 has to be replaced. From the column of "*A*" in Table 4, we can identify that the objects *B*, *D*, and *F* have to be added to the *DQ* along the object *A*, and obtain the current status of the *DQ* as shown in Figure 5. Object *F* is the first retrieved from the *DQ* for disassembly process. Because no object interferes object *F* as seen from the column "*F*" of Table 4, dismantle the fasteners of object *F* if any, and remove object *F* from object *A*.

The Relationship Matrix is reduced after the removal of object *F* as shown in Table 8. The next object in the *DQ*, object *D*, cannot be removed directly because *D* is interfered by *E* that is then added to the *DQ*. Continue the disassemblability check of each object in the *DQ* until the target object *A* is finally retrieved. According to the proposed inference scheme, the minimum disassembly steps to retrieve object *A* can be obtained as follows:

(1). Unscrew the Fastening Screws of object F, and remove object F from object A.
(2). Remove object E from object B.
(3). Unscrew the Fastening Screw of object D, and remove object D from object B and object A.
(4). Unscrew the Fastening Screws of object B, and remove object B from object A.
(5). Remove the target object A.

Noted that object *C* is not in the *DQ* because *C* does not interferes the removal of *B* that can be observed from the Relationship Matrix. Therefore, object *C* is removed along with object *B* in disassembly step 4 without disconnecting *C* and *B*. The reassembly sequence after the service is simply the reverse order of the derived disassembly sequence.

F
D
B
A

Figure 5 The status of the *DQ* of the example layout after step (2)

Table 8

The reduced Relationship Matrix after the removal of object *F*

$$RM_{r2} = \begin{array}{c} \text{R-ing} \to \\ \text{R-ed} \downarrow \\ A \\ B \\ C \\ D \\ E \end{array} \begin{array}{c} \\ \left[\begin{array}{ccccc} A & B & C & D & E \\ 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 2 & 3 & 0 & 0 & 0 \\ 0 & 2 & 1 & 1 & 0 \end{array}\right] \end{array}$$

6. Applications

The proposed structure representation and the inference rules can be readily applied to the design of an automated analyzer for product assembly and disassembly. This session describes a prototype software tool, *DASER*, for product assembly planning and service mode analysis. We use MS Visual Basic for the interface design and MATLAB for the kernel of the automatic sequencing of assembly and disassembly. MATLAB is selected because the capability of matrix manipulation. *DASER* applies the proposed scheme to generate the assembly and disassembly steps required in the layout analysis.

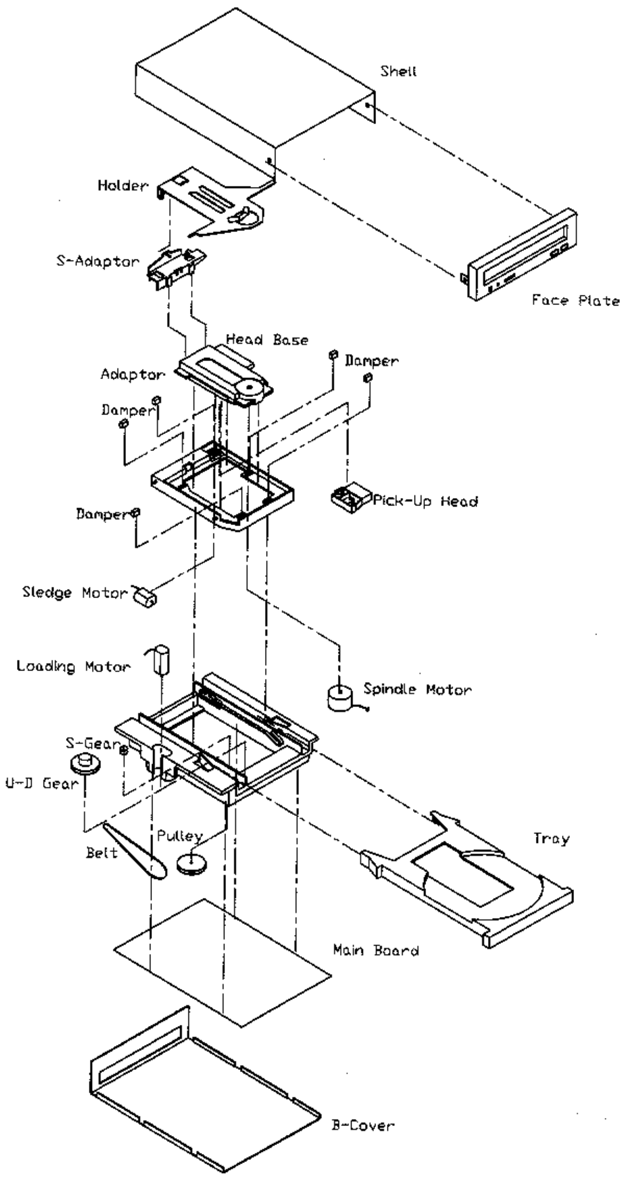


Figure 6 The explosive view of a CD ROM

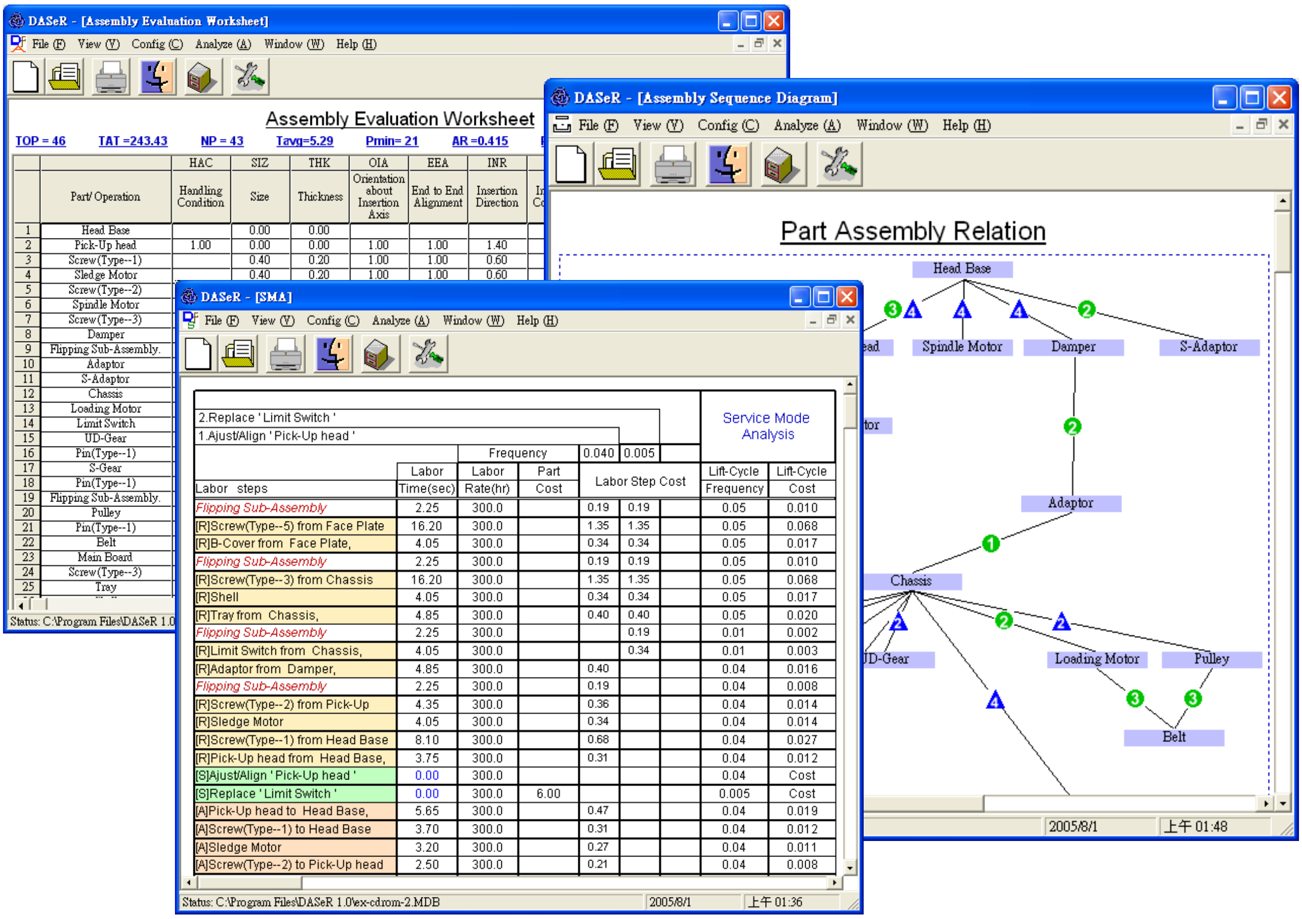


Figure 7 The sample output of the assembly and disassembly analyzer, *DASER*

Users first input the product architecture via an interactive interface. The example of a CD ROM as shown in Figure 6 can then be represented as the OR-Graph as shown in Figure 7. The assembly planning refers to the Modified Westinghouse Method [11] for the assemblability evaluation. The Modified Westinghouse method estimates the assembly time for each assembly action from a parameterized spreadsheet. The software tool can automatically generate a feasible assembly sequence to estimate the total assembly time of the product.

On the other hand, the Service Mode Analysis [8] analyzes the service mode cost from the product of the total labor step cost and the service mode frequency for a particular service mode. The estimation of the labor cost of the service mode requires the derivation of the disassembly and reassembly steps. The software tool can provide the minimum assembly and disassembly sequence for the specified parts in SMA. The sample output of assemblability evaluation and SMA are shown in Figure 7.

7. Conclusions

To facilitate the concurrent engineering at the early stage of design, a structure representation of the product design is essential. This paper presents the Relationship Matrix to describe the assembly architecture of a fixed-form product. The Relationship Matrix describes both the information of the component connectivity and the layout precedence, and considers only the functional elements to greatly reduce the searching space. The proposed methodology provides a simple solution for the concurrent analysis of assembly and disassembly at the early stage of design when the detailed geometry of product constituents is not yet available. The mandatory subassemblies can be identified, and feasible assembly sequences and the minimum service steps for the malfunction component can be easily derived. The structure representation and the inference kernel can be applied to concurrent design review for assemblability, serviceability, and recyclability as illustrated by the prototype software tool. However, the memory requirement and the searching efficiency using the matrix representation for a complex system will become a concern. The applicability of the proposed scheme to complex products is pending future investigation. The introduction of Entity Relationship models, objected oriented data structure, and data standardization scheme such as STEP and PDML will be possible solutions to improve data interchangeability and searching efficiency.

Acknowledgments

The authors would like to thank the National Science Council of the Republic of China for partially supporting this research under Contract No. NSC 88-2212-E-011-017.

References

[1]. Boothroyd, G. & Dewhurst, P. (1983). *Design for Assembly: a designer's handbook*, Boothroyd Dewhurst Inc., Wakerfield, Rhode Island.

[2]. Chen, R. W., Navin-Chandra, D., & Prinz, F. B. (1993). Product Design for Recyclability: A Cost Benefit Analysis Model and its Application. *Electronics and the Environment*, IEEE, 178 – 183.

[3]. Clausing, D. (1994). *Total Quality Development: A Step-by-Step Guide to World-Class Concurrent Engineering*, New York: ASME press.

[4]. De Fazio, T. L. & Whitney, D.E. (1987). Simplified Generation of All Mechanical Asembly Sequences. *IEEE J. of Robotics and Automation*, RA-3 (6), 640-658.

[5]. Dini, G. & Santochi, M. (1992). Automated Sequencing and Subassembly Detecting in Assembly Planning. *Annals of The CIRP*, Vol.41, pp.1-4

[6]. Homem de Mello, L.S. & Sanderson, A.C. (1990). AND/OR graph representation of assembly plans. *IEEE Transactions on Robotics and Automation*, 6(2), 188 – 199.

[7]. Huang, Y. M. & Huang, C.-T. (2002). Disassembly matrix for disassembly processes of products. *Int. Journal of Production Research*, 40(2), 255-273.

[8]. Ishii, K., Eubanks, C.F., & Marks, M. (1993). Evaluation Methodology for Post-manufacturing Issues in Life-cycle Design. *Concurrent Engineering: Research and Application*, 1, 61-68.

[9]. Kuo, T.C., Zhang, H.C., & Huang, S.H. (2000). "Disassembly analysis for electromechanical products: A graph-based heuristic approach. *Int. Journal of Production Research*, 38(5), 993-1007.

[10]. Ong, N.S. & Wong, Y.C. (1999). Automatic Subassembly Detection from a Product Model for Disassembly Sequence Generation. *Int. J. of Advanced Manufacturing Technology*, 15, 425-431.

[11]. Struges, R.H. & Kilani, M.I. (1992). Towards an Integrated Design for an Assembly Evaluation and Reasoning System, *Computer Aided Design*, 24(2), 67-79.

[12]. Tang, Y., Zhou, M. C., Zussman, E., & Caudill, R. (2000). Disassembly modeling, planning and application: a review. *Proc. - IEEE Int. Conf. on Robotics and Automation*, Vol. 3, (pp. 2197 – 2202)