INTERNATIONAL DESIGN CONFERENCE - DESIGN 2004
Dubrovnik, May 18 - 21, 2004.

DESIGN 2004

PRODUCT DISASSEMBLY SEQUENCES APPROACH IN THE EARLY STAGES OF PRODUCT DESIGN

M. Kljajin and M. Opalić

Keywords: product structuring, design for disassembly, optimisation techniques

1. Introduction

The objective of the work described in this paper is to establish procedures, which can be used to assess servicing costs at the earliest stages of product design. It is intended that the procedures will build upon the work in product design for assembly (DFA) [Boothroyd & Dewhurst 1985, Kljajin 1998, Kljajin 1999], which is being used by many industries in Europe and the USA and has had numerous reported successes. The power of the DFA methodology arises from us ability to quantify assembly costs at the earliest stages of product design, and then establish an efficiency rating that indicates scope for possible design improvements. A similar metric for serviceability is the goal of the current research. This metric will be useful in establishing an overall rating of the assembly with respect to its expected lifetime servicing costs. A preliminary literature review was conducted to establish a definition for serviceability [Jones 1988, Cralley et al 1990, Kljajin 2000]. From this references review it became clear that definitions of serviceability are largely dependant on the particular industry where they are used. Further, some of the definitions include consideration of field data which may be difficult to obtain or unavailable at the initial design stages, especially when dealing with an entirely new design. A single set of guidelines and metrics cannot therefore be completely applied to all sections of the industry. However one common thread that is implicit in most of the definitions and guidelines is the ease of disassembly of the service items. It seems that it may be possible to establish metrics or this characteristic at the earliest stages of product design. To achieve this goal the following are needed (a) A model of the assembly to represent the relations between parts. (b) An algorithm working on the model to generate the optimal disassembly sequences of parts for removal of service items or for recycling at the end of product life. (c) A procedure to expand the sequence of parts determined in step (b) to include additional disassembly operations: e.g. melting of solder joints, etc. (d) Suitable metrics based on disassembly and reassembly times for identified service items. In this paper is described the work, which has been carried out in steps (a) and (b) above.

2. Choice of assembly model

The trend in the literature for representing the relationships in a mechanical assembly has been to move away from precise geometric descriptions towards more general part/object oriented representation schemes. Several such schemes to represent mechanical assemblies were studied [Sanderson et al 1986, 1988]. Amongst these the relational model described by Sanderson at al [Sanderson et al 1988] was found to be the most suitable representation of an assembly/disassembly for our work. Sanderson et al [Sanderson et al 1986, 1988] have also described an algorithm to generate all the possible assembly/disassembly sequences from the relational model. However no procedures have been found in the literature to generate disassembly sequences for specific parts. This is the subject of our work. For this purpose, the relational model of Sanderson at al [Sanderson et al 1988] was used with some extensions. This extended model is described below.

2.1 Relational model

This model consists of four entities: Parts, Contacts, Attachments and Relations. The parts are identified by part names, while unique numbers identifies the attachments, contacts and relations. Parts are the discrete solid objects, which make up an assembly. The mating configurations in the assembly are represented by contacts. Contacts can be of the following types: (a) Planar contact (e.g. two blocks placed one on top of the other). (b) Cylindrical contact (e.g. round peg in cylindrical hole). (c) Polygonal contact (e.g. square peg in a square hole, or any other sectional shape which completely restricts rotation). (d) Slotted contact (e.g. the contact between the piston and the small end of an internal combustion engine connecting rod). (e) Threaded cylindrical contact.

These contacts place different constraints on the contacting parts. Each contact relates exactly two parts. Attachments are entities used to represent the binding of the contacts. Attachments have one or more targets and one agent. Target refers to the contacts that are secured and agent refers to the entity that is responsible for the binding. For example, in the case of a screw and nut with a washer in between, the agent is the threaded cylindrical contact between the screw and nut, and the targets are the planar contacts between screw + washer and washer + nut. The targets are always contacts and the agent can be either a contact or a part. Four attachment types are used in the current version of the model: screw attachment, press fit attachment, glue attachment, and clip attachment. This preliminary list, which covers many assembly situations, has been chosen for development purposes. The list can be readily expanded.

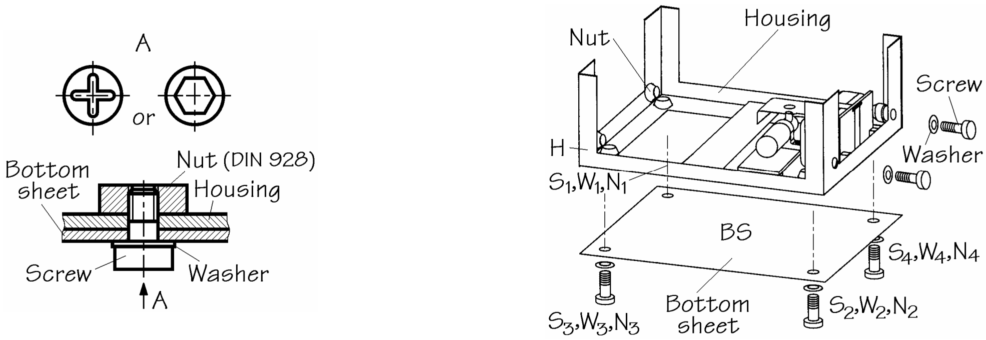

Figure 1. Screw/washer/nut assembly (left) and example for its applying (right)

Relations are entities used to form the associations between the parts, contacts and attachments in the model. They are non-directional and are of the following types: part contact relation, target attachment relation, and agent attachment relation. Relations get their name from the entities they connect. The part contact relation is used to relate a part to its contact numbers. The target attachment relation is used to relate an attachment to its targets. The agent attachment relation is used to relate an attachment to its agent. The use of the relational model to represent assemblies is illustrated with the example shown in Fig. 1 (Notices: The bottom sheet has the role of the second washer, and the nut is a component of housing. The best variant for assembly/disassembly is crosshead or hexagonal socked head screw. This is very often in use by household appliances.). Fig. 2 shows the relational model for the screw/washer/nut assembly shown in Fig. 1. The relational model can be considered as a set of parts, contacts and attachments, relates by the connections as shown. The screw has a threaded cylindrical contact with the nut, which forms the agent of a screw attachment A_1. The attachment A_1 binds the planar contacts between screw + washer, washer + bottom sheet (it is similar as washer) and bottom sheet + nut (which is part of housing). These relations are labelled AAR_1, TAR_1 and TAR_2. The entities in Fig. 2 stand for: CC_1, CC_2 – cylindrical contacts; PC_1, PC_2, PC_3 – planar contacts; TCC_1

– threaded cylindrical contact; SA_1 – screw attachment; AAR_1 – agent attachment relation; TAR_1, TAR_2, TAR_3 – target attachment relations; PCR_1 through PCR_{12} – part contact relations.

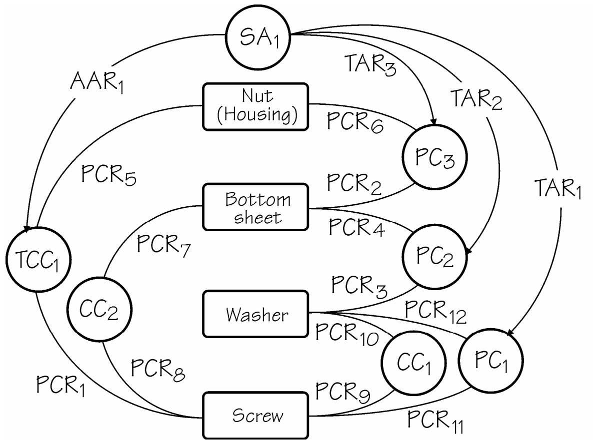

Figure 2. Relational model for screw/washer/nut assembly

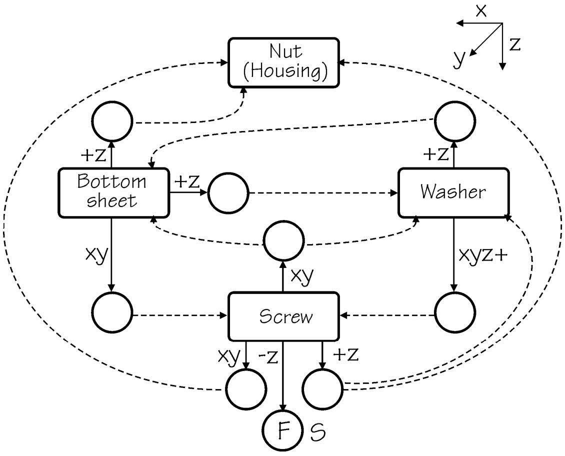

Figure 3. Disassembly diagram for screw/washer/nut assembly

It must be noted, that although the target attachment and agent attachment relations are shown directed, to enhance understanding, they are in fact non-directional. The input to a computer program could be consists of a description of the types of contacts in the assembly along with the disassembly directions but this is not subject of analysing in this paper.

3. The disassembly problem

A representation of the assembly suitable for the generation of disassembly sequences is described in this section. This representation is termed the disassembly diagram (DAD). Before laying out a description of the DAD it may be worthwhile to review some of the literature cited earlier. Two schools of thought can be discerned in the representation of mechanical assemblies. The geometric/relational representation, [Sanderson et al 1986, Sanderson et al 1988], and the approach based on precedence knowledge alone. It seems clear that precedence is the most logically complete description of the assembly that can be used for sequence generation. However, the methods cited attempt to derive this precedence knowledge by querying the user. This can help to enhance the users understanding of the product structure. However, for anything other than simple assemblies the problem of defining the precedence knowledge becomes a cumbersome one.

3.1 Local disassembly

Local disassembly refers to the process of removing a part with respect to its local constraints. Local constraints refer to the relative constraints between various parts in direct contact, or between parts that would immediately obstruct each other's removal. This information is represented in terms of the disassembly directions of parts with respect to their contacts or obstructions. The term "disassembly direction" refers to the possible directions of motion for a part with respect to any of its contacts. The user describes these as each pan is added to the assembly. For many types of contacts there are very few feasible motions between the contacting parts. For example a cylindrical pin in a hole can translate in the positive or negative direction along the axis of the pin and rotate about the axis. With this in mind each direction of relative motion is represented as a four-element list (A-F_1-F_2-T), where A is the axis name, say x, y or z; F1 is the translational degree of freedom, which can take values +1 or -1; F2 is the rotational degree of freedom, which can take the values +1, -1, 0 or F, (The value 0 indicates that rotation is unrestricted, and F means the part has to be removed without rotation); T is a number such that coaxial directions will carry the same number, and parallel directions that are not coaxial will have different numbers. In general the disassembly directions will comprise one or more of these four-element list. The problem of generation of disassembly sequences for a part in an assembly consists of

the following tasks (a) Freeing the part of all attachments. (b) Finding the succeeding part in the disassembly sequence. (c) Disassembly of the succeeding part.
This approach was used in a heuristic search procedure with some success, by the authors [Subramani & Dewhurst 1990]. However the procedure, due to its reliance on heuristics alone, failed to extract the optimal sequence of disassembly for other than simple assemblies. For this reason an improved procedure using an algorithmic approach through the use of the disassembly diagram was developed as explained below.

3.2 The disassembly diagram

The disassembly diagram is a model for the assembly that facilitates straightforward generation of optimal disassembly sequences for all the parts in the assembly through the application of a systematic procedure. The model contains knowledge about the disassembly precedence of various parts in the different directions of disassembly. The term direction here refers to the six positive and negative vectors along the three coordinate axes x, y and z. The diagram will be illustrated with a simple assembly.
The following notations are used: *Rectangles:* Parts; *Solid edge:* Possible disassembly directions; *Dotted edges:* Represent contacts which must be broken for removal of parts along disassembly directions. A cost is associated with each dotted edge; *Circles:* Nodes, which are used to keep track of the costs, as contacts are broken; *Circles labelled F:* Free nodes, which indicate parts immediately available for removal; *Free parts:* Parts immediately available for removal. These nave free nodes associated with them.
There is only one solid edge connection leading into each node and one or more dotted edges leading out. There can be several solid and dotted edges leading out and into a part. Nodes labelled F indicate that the associated part is free in the direction linking the node and the part. Parts have associated with them, a list of all the attachments that would have to be broken to free the part and a time associated with each attachment. Breaking the attachments is mandatory for freeing a part hence no separate entity is required to represent the attachment on the disassembly diagram. The disassembly diagram is illustrated in Fig. 3 for the example shown earlier in Fig. 1. The parts screw is indicated as free in the z+ and z- directions. To generate the DAD from the relational model two main procedures were used namely: Set-list procedure and List-contacts procedure.
Set-list returns the minimum number of unique disassembly directions along which a part can be moved with respect to its neighboring parts. In many cases parts will be totally constrained and set-list returns ail the possible directions for disassembly. This is because unless there is a direction or directions that would allow me removal of a part without affecting the neighboring pans, all possible directions should be considered for disassembly. In this way, as other parts are removed possible disassembly directions will not be overlooked. For example the procedure would return as possible directions the list ((z -1 -1)) for the screw in example in Fig. 1 and the possible list of directions as ((x 1 F)(x -1 F)(y 1 F)(y -1 F)(z 1 0)(z -1 0)) for washer in the same example.
List-Contacts forms a list of all the contacts that would need to be broken in a given direction. With this information the parts that need to be removed can quite easily be determined. For example to move the washer in the z+ direction the planar contact between screw and washer has to be broken, that is the screw has to be removed. Applying these procedures, to each part in turn results in the generation of the DAD.

3.3 The Disassembly Algorithm

To start with, the entire assembly is modelled by the DAD with parts immediately available for removal indicated by free nodes. The free nodes have a cost for removal associated with them. As "free" parts are removed the disassembly diagram is progressively reduced and the other free nodes are generated. These subsequent free nodes contain the cost of removing the associated part plus the cost of removing the preceding parts, which resulted in the node becoming free. The goal is reduce the disassembly diagram to a state where the part, that needs to be removed, has a free node associated with it. This is done in a branch and bound search algorithm [Winston 1984] by removing one part at a time and computing the list of part names and the sum of disassembly times. Finally, when the part to be removed is free, its free node contains information about the disassembly sequence of parts and the total disassembly time. At this stage the procedure terminates. If at any point more than one free node is available, then the one with the least cost is selected each time. If one or more have the same value then the nodes are chosen at random.

This procedure is illustrated by an example of a simple roller assembly shown in Fig. 4. Lets assume that the roller (R) is the part needing service. This is held in place by two bearings (B_1 and B_2), and four screws (S_1, S_2, S_3 and S_4) securing them down to the holder (H). For simplicity, dotted edges carry a cost penalty of 2 units and the screw attachments carry a cost penalty of 5 units.

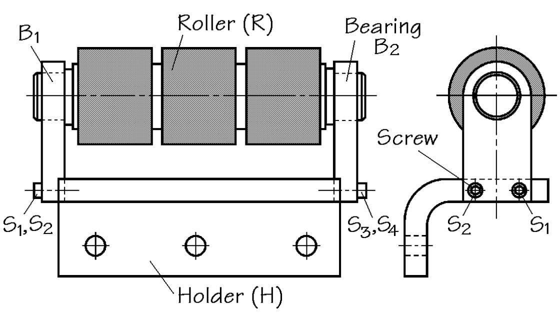

Figure 4. Roller assembly

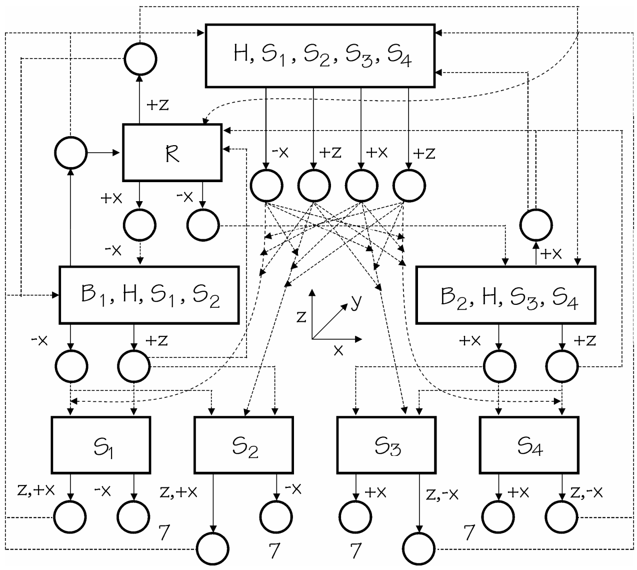

Figure 5. Initial state of disassembly diagram

The progress of the search procedure is illustrated in Fig. 5 all free nodes have a cost of 5. The information about attachments is stored in the rectangles in the following manner: each rectangle contains the list of attachments, which act upon the contacts of the associated part with the adjacent parts. When a part is freed the cost on the free node is added up to all the nodes that are connected to this part by dotted edges. To start with all the free nodes contain the cost of breaking the attachments binding the contacts of the corresponding free part and the cost of removing tree part. As parts are removed, contacts (dotted edges) are broken. This could either generate additional free nodes or add cost to some of the existing nodes. In the latter case the cost on the free node associated with the part removed is added. In the former case the cost of detaching and removing the just freed part is also added. Thus, two different situations arise when a "free" part is removed. The way in which the costs are added to the remaining nodes that are connected by dotted edges to the "free" part depends on whether or not the nodes become "free". This is expressed symbolically by formulae (1) and (2) below. Let the number of parts be n and the number of nodes be c. Let P_j represent parts $1 \le j \le n$ and N_k represent nodes, $1 \le k \le c$. Let P_j be free to be removed and N_k be the corresponding free node. Let $AC(P_j)$ be a function that returns the cost of breaking the attachments on P_j. Let $CC(P_j)$ be a function that returns the cost of breaking the contacts on P_j. Let $NC(N_k)$ be a function to compute the cost on node N_k. Let N_m be such that there is a dotted edge connecting N_m and P_j. If N_m does not become free as a result of removing P_j then

$$NC(Nm) = NC(Nm) + NC(Nk) \tag{1}$$

If N_m becomes free as a result of removing P_j then

$$NC(Nm) = NC(Nm) + NC(Nk) + AC(Pj) + CC(Pj) \tag{2}$$

where P_j is the part freed as result of N_m becoming free.

Using the procedure described by formulae (1) and (2) to perform cost additions to the nodes, the disassembly diagram is reduced from the initial state in Fig. 5 to the new states, removing one "free" part every time.
It is to be noted that the branch and bound strategy is guaranteed to generate an optimal path subject to separate considerations of the last step. The last step here refers to the breaking of contacts that surround the service part, P_s. That is it corresponds to the cost returned by the function $CC(P_s)$. In this paper alt the contacts are assumed to have a constant cost penalty and the algorithm described here is guaranteed to generate optimal disassembly sequences under these conditions. However, in general it will always be necessary for the algorithm to explore every possible next step in the disassembly diagram in order to ensure the true optimum path.

4. Conclusions

This paper describes a methodology for modelling mechanical assemblies and an algorithm for the generation of optimal sequences for disassembly. This algorithm is intended to form the basis for the assessment of service and maintenance difficulties in product design. Further work in this project will concentrate on the establishment of a service time database and a service efficiency metric. In addition the present procedure goes not consider additional time penalties, which may be associated with the simultaneous breaking of several contacts. The implication of this on the selection of optimal disassembly sequences will also be the subject of future research of disassembly and dismantling during the recycling process.

Acknowledgement

Presented results have been raised under the programme TEST supported by the Ministry of Science and Technology (Republic of Croatia) (Project of Modular System for Recycling of WEEE, no. TP-02/0120-14).

References

Boothroyd, G. & Dewhurst, P., Design for Assembly Handbook, University of Rhode Island, 1985.

Jones, J. V., Engineering Design, Reliability, Maintainability and Testability, TAB Professional and Reference Books, Division of TAB BOOKS Inc., Blue Rage Summit, PA, 1988.

Cralley, W. E. et al, Computer Support for Conducting Supportability Trade-offs in a Team Seeing, Institute for Defence Analysis, IDA paper P-2313, 1990.

Kljajin, M., Design for Assembly, Tehnički vjesnik/Technical Gazette, Vol. 5, No. 3/4, 1998, pp 37-44.

Kljajin, M., Design for Disassembly, Tehnički vjesnik/Technical Gazette, Vol. 6, No. 1/2, 1999, pp 41-48.

Kljajin, M., Design for Disassembly due to Maintenance and Recycling, Proceedings Recent Advances in Design for Manufacture (DFM), DE-Vol. 109, Ed. B. S. Kim, ASME International, New York, 2000, pp 51-59.

Sanderson, A. C. et al, Automatic Generation of Mechanical Assembly Sequences, Technical Report, CMU-RI-TR-88-19, 1988.

Sanderson, A. C. et al, AND/OR Graph Representation of Assembly Plans, Technical Report, CMU-RI-TR-86-8, 1986.

Subramani, A. K. & Dewhurst, P., Preliminary Investigations on Design for Service, International Forum for Design for Manufacturability, Newport, Rhode Island, 1990, pp14-15.

Winston, P. A., Artificial Intelligence, Addison Wesley, Second Edition, Reading, Massachusetts, 1984.

Milan Kljajin
University of Osijek, Faculty of Mechanical Enginering
Trg Ivane Brlić-Mažuranić 18, Slavonski Brod, Croatia
Telephone: +385 35 446 188, Telefax: +385 35 446 446
E-mail: mkljajin@sfsb.hr