

January 01, 2001

## Computational algorithm to evaluate product disassembly cost index

Ibrahim Zeid  
*Northeastern University*

Surendra M. Gupta  
*Northeastern University*

---

### Recommended Citation

Zeid, Ibrahim and Gupta, Surendra M., "Computational algorithm to evaluate product disassembly cost index" (2001).. Paper 29.  
<http://hdl.handle.net/2047/d10003186>



Laboratory for Responsible Manufacturing

## Bibliographic Information

Zeid, I. and Gupta, S. M., "Computational Algorithm to Evaluate Product Disassembly Cost Index", ***Proceedings of the SPIE International Conference on Environmentally Conscious Manufacturing II***, Newton, Massachusetts, October 28-29, pp. 23-31, 2001.

## Copyright Information

*Copyright 2001, Society of Photo-Optical Instrumentation Engineers.*

*This paper was published in Proceedings of SPIE (Volume 4569) and is made available as an electronic reprint with permission of SPIE. One print or electronic copy may be made for personal use only. Systematic or multiple reproduction, distribution to multiple locations via electronic or other means, duplication of any material in this paper for a fee or for commercial purposes, or modification of the content of the paper are prohibited.*

## Contact Information

Dr. Surendra M. Gupta, P.E.  
Professor of Mechanical and Industrial Engineering and  
Director of Laboratory for Responsible Manufacturing  
334 SN, Department of MIE  
Northeastern University  
360 Huntington Avenue  
Boston, MA 02115, U.S.A.

(617)-373-4846 **Phone**  
(617)-373-2921 **Fax**  
gupta@neu.edu **e-mail address**

<http://www.coe.neu.edu/~smgupta/> **Home Page**

# A Computational Algorithm to Evaluate Product Disassembly Cost Index

Ibrahim Zeid and Surendra M. Gupta  
Laboratory for Responsible Manufacturing  
334 SN, Department of MIME  
Northeastern University  
360 Huntington Avenue  
Boston, MA 02115

## ABSTRACT

Environmentally conscious manufacturing is an important paradigm in today's engineering practice. Disassembly is a crucial factor in implementing this paradigm. Disassembly allows the reuse and recycling of parts and products that reach their "death" after their life cycle ends. There are many questions that must be answered before a disassembly decision can be reached. The most important question is economical. The cost of disassembly versus the cost of scrapping a product is always considered. This paper develops a computational tool that allows decision-makers to calculate the disassembly cost of a product. The tool makes it simple to perform "what if" scenarios fairly quickly. The tool is Web based and has two main parts. The front-end part is a Web page and runs on the client side in a Web browser while the back-end part is a disassembly engine (servlet) that has disassembly knowledge and costing algorithms and runs on the server side. The tool is based on the client/server model that is pervasively utilized throughout the World Wide Web. An example is used to demonstrate the implementation and capabilities of the tool.

**Keywords:** Design for Disassembly, Algorithms, Disassembly Cost Index

## 1. INTRODUCTION

In recent years product designers have been under obligation to consider the environmental impact of their product designs. Such obligations have led to the emergence of design initiatives that address the environmental concerns of product life cycle and production processes. Not only must the designs of products satisfy functional specifications, but they should also be easy to assemble, disassemble and possess a host of other attributes. Lately, there have been efforts to emphasize environmental considerations into the design of products, a concept known as the *Green Design*.

Even though many companies were skeptical to embrace the green design concept assuming that it would be enormously expensive, the current consensus is that, with proper design, not only is green design more cost effective, in many cases, it could actually generate a positive income. Besides this, it is necessary because of competition, consumer demand and the prevailing laws.

This paper builds on the technique recently reported by Veerakamolmal and Gupta to analyze the design efficiency of a product to study the effect of EOL disassembly and disposal on the environment<sup>16</sup>. The design efficiency is measured using a Design for Disassembly Index (DfDI). DfDI uses a disassembly tree (DT) which relies on the product's structural blueprint<sup>18</sup>. The DT can be used to identify precedent relationships that define the structural constraints in terms of the order in which components can be retrieved. The development of this index involves the analysis of a logic disassembly table to find the combination of components and materials together with their layout in the product so as to provide the optimum cost-benefit ratio for end-of-life retrieval. The cost considerations in this analysis include disposal and disassembly costs, while the benefit is derived from the sales of recovered components and materials in terms of reuse and recycling revenue.

This paper develops a computational tool that allows decision makers to calculate disassembly cost of a product. The tool also makes it feasible to perform “what if” analyses in a fairly simple and quick way. The tool is Web based and has two main parts. The front-end part is a Web page. The back-end part is a disassembly engine (servlet) that has disassembly knowledge and costing algorithms. The tool is based on the client/server model that is pervasively utilized throughout the World Wide Web. The front-end runs on the client side in a Web browser. The back-end runs on the server side. This paper discusses the implementation of such a computational tool and presents sample examples.

## 2. LITERATURE REVIEW

Among other things, a good product design curtails manufacturing costs, reduces lead times for production, and at its end-of-life enables the recovery of valuable materials, diminishes environmental detriments and helps ease product disassembly as well as component removal.

Numerous analytical tools have been developed to assist and/or evaluate different aspects of product design<sup>17</sup>. Ishii *et al.*<sup>6</sup> developed a methodology to design a product for retirement using a hierarchical semantic network that consists of components and subassemblies. Navin-Chandra<sup>13</sup> presented an evaluation methodology for Design for Disassembly (DfD). He developed a software called ReStar, which optimizes the component recovery plan. Subramani and Dewhurst<sup>14</sup> investigated procedures to assess service difficulties and their associated costs at the product design stage. They used a metric for serviceability to assess an overall rating of a product’s design in comparison to its expected lifetime servicing costs.

Isaacs and Gupta<sup>5</sup> have developed an evaluation methodology that enables an automobile designer to measure disassembly and recycling potential for different automobile designs. With respect to the designs, the method uses Goal Programming to analyze the tradeoff between the profit aspiration levels of the disassembler and the shredder. Johnson and Wang<sup>7</sup> used a disassembly tree (DT) in designing products to enhance material recovery opportunities. Their technique employs selective disassembly of a product by performing a profit-loss analysis on various combinations of components. Vujosevic *et al.*<sup>19</sup> have studied the design of products that can be easily disassembled for maintenance.

Veerakamolmal and Gupta<sup>16</sup> have advanced a technique for analyzing the design efficiency of electronic products, in order to study the effect of end of life (EOL) disassembly and disposal on the environment. The design efficiency is measured using a Design for Disassembly Index (see the next section for an overview). The index offers designers with an important measure to help improve the future products.

For additional literature on disassembly and recycling, see Moyer and Gupta<sup>12</sup>, Lee *et al.*<sup>10</sup> and Gungor and Gupta<sup>2</sup>.

## 3. DESIGN FOR DISASSEMBLY INDEX (DfDI)

It is not uncommon for a designer to be faced with a dilemma of choosing among two or more design alternatives. A product design can make an enormous difference in the product’s retirement strategy. To compare the merits of two (or more) designs from the disassembly point of view, we must compare the cost-benefit of disassembly from one design against the others. The cost-benefit function ( $Z$ ) consists of four terms, (viz., total resale revenue ( $TRR$ ), total recycling revenue ( $TCR$ ), total processing cost ( $TPC$ ), and total disposal cost ( $TDC$ )) as follows:

$$Z = TRR + TCR - TPC - TDC \quad (1)$$

$TRR$  is directly influenced by  $RV_j$  and  $TC_i$ .  $RV_j$  is the resale value of component  $P_j$ , and  $TC_i$  is the cost per unit of acquiring and transporting product  $i$  from the distribution centers (or collection sources) to the disassembly facility. The revenue equation represents revenue less the cost of product acquisition, which can be formulated as

$$TRR = \sum_{j \in P_j \in LS^S(Root_i)} (RV_j \cdot \{Q_{ij}\} \cdot \{X_{ij}\}) - TC_i \quad (2)$$

where  $LS^S(Root_i)$  is the set of selected leaf successors of the root node in product  $i$ ,  $Q_{ij}$  is the multiplicity matrix representing the number of each type of component  $P_i$  obtained from each type of product  $i$ ,  $X_{ij}$  is the matrix

representing the (mutually exclusive combination) selection of component  $P_j$  retrieved from product  $i$  for reuse ( $X_{ij} = 1$ ) or recycle and/or disposal ( $X_{ij} = 0$ ) and  $\{\beta_{ij}\}$  is element in row  $i$  and column  $j$  of matrix  $\beta_{ij}$ .

$TCR$  is calculated by multiplying the component recycling revenue factors by the number of component units recycled for materials content as follows:

$$TCR = \sum_{j \in P_j \in LS^S(Root_i)} (CI_j \cdot DW_j \cdot CRP_j \cdot \{Q_{ij}\} \cdot (1 - \{X_{ij}\})) \cdot CF \quad (3)$$

Note that each component has a percentage of recyclable contents ( $CRP_j$ ) (the portion not recycled must be properly disposed of).  $CI_j$  is the recycling revenue index (varying in value from 1 to 10) representing the degree of benefit generated by the recycling of component  $P_j$  (the higher the value of index, the more profitable it is to recycle the component),  $DW_j$  is the weight of the component, and  $CF$  is the recycling revenue factor (\$/unit of index scale).

$TPC$  can be calculated from the process makespan ( $TD_i^s$ ) and the processing cost per unit time ( $PC$ ) as follows:

$$TPC = TD_i^s \cdot PC \quad (4)$$

and, in turn,  $TD_i^s$  (assuming that there is a demand for all selected components) can be obtained using the following equation:

$$TD_i^s = \left( \underset{\forall P_j \in LS^S(Root_i)}{Max} \lceil \{X_{ij}\} \rceil \right) \left( T(Root_i) \right) + \sum_{k=1}^{S_i} \left\{ \left( \underset{\forall P_j \in LS^S(A_{ik})}{Max} \lceil \{X_{ij}\} \rceil \right) \left( T(A_{ik}) \right) \right\} \quad (5)$$

where  $LS^S(A_{ik})$  is the set of selected leaf successors of subassembly node  $k$  in product  $i$ ,  $T(Root_i)$  is the time to disassemble root node of the product  $i$  (unit time),  $T(A_{ik})$  is the time to disassemble subassembly  $k$  from product  $i$  (unit time) and  $\lceil \alpha \rceil$  gives the smallest integer that is larger than or equal to  $\alpha$ .

$TDC$  is calculated by multiplying the component disposal cost by the number of component units disposed as follows:

$$TDC = \sum_{j \in P_j \in LS^S(Root_i)} (DI_j \cdot DW_j \cdot (1 - CRP_j) \cdot \{Q_{ij}\} \cdot (1 - \{X_{ij}\})) \cdot DF \quad (6)$$

Note that  $DI_j$  is the disposal cost index (varying in value from 1 to 10) representing the degree of nuisance created by the disposal of component  $P_j$  (the higher the value of index, the more nuisance the component creates and hence it costs more to dispose it of), and  $DF$  is the disposal cost factor (\$/unit of index scale).

#### 4. DISASSEMBLY COMPUTATIONAL ALGORITHM

The following algorithm incorporates the related equations described in the previous section. It leads to the calculation of DfdI in the following steps:

- Step 1:** List for each component, its ID, predecessor, resale value ( $RV_j$ ), multiplicity ( $\{Q_{ij}\}$ ), weight ( $DW_j$ ), recyclable percentage ( $CRP_j$ ), recycle index ( $CI_j$ ), and disposal index ( $DI_j$ ).
- Step 2:** Generate the mutually exclusive combination matrix ( $X_{ij}$ ) for component(s) selection.
- Step 3:** Obtain the cost of acquiring each product ( $TC_i$ ), processing cost/unit time ( $PC$ ), disassembly times

( $T(\text{Root}_i)$  and  $T(A_{ik})$ ), recycling revenue factor ( $CF$ ), and disposal cost factor ( $DF$ ). Calculate  $TRR$ ,  $TCR$ ,  $TPC$  and  $TDC$  with respect to the selection of reused components in each of the corresponding mutually exclusive combination.

**Step 4:** Calculate total benefit ( $TRR + TCR$ ), total cost ( $TPC + TDC$ ), DfDI (total benefit/total cost) and net benefit ( $Z$ ) for each combination.

As can be seen from the above steps, the input to the algorithm is the disassembly data including the disassembly tree, the resale value  $RV_j$  of a component, the multiplicity  $Q_{ij}$ , the weight  $DW_j$ , the recycle percentage  $CRP_j$ , the recycle index  $CI_j$ , the disposal index  $DI_j$ , cost of acquiring each product  $TC_i$ , processing cost/unit time  $PC$ , disassembly times  $T(\text{Root}_i)$  and  $T(A_{ik})$ , recycling revenue factor  $CF$ , and disposal cost factor  $DF$ . The algorithm permits a designer to choose none, one, more than one or all the components to be disassembled and retrieved. By generating all mutually exclusive combinations of selected components, matrix  $X_{ij}$  is built. The output from the algorithm include the DfDI and the net benefit for each mutually exclusive combination of selected components that allow product designers to evaluate their designs from a disassembly point of view. These figures help to decide whether to reuse or dispose of a component.

A schematic diagram of the algorithm is shown in figure 1. The algorithm receives product disassembly data from a Web page. The algorithm proceeds by calculating the various quantities described above, including the disassembly index (DfDI). The algorithm displays the results in the form of a Web page for evaluation and analysis.

## 5. WEB-BASED DISASSEMBLY TOOL

We have developed a Web-based disassembly tool that implements our disassembly computational algorithm. The tool is based on the popular client/server architecture that is used extensively for Web-based applications<sup>20</sup>. We use two-tier client/server architecture as shown in figure 2. The disassembly computation algorithm engine resides on the server while the front end (Web page) of the algorithm is accessible via any client.

The disassembly database holds all the disassembly input data as well as the disassembly trees of different products. This database is used by the disassembly algorithm to calculate the various disassembly cost components, which ultimately lead to calculating the design for disassembly index. The communication between the client and the server is achieved via using the Java Servlet technology<sup>3,4,8</sup>. A servlet has been developed and stored on the server. A call to the server is embedded in the disassembly Web page. When the user sends a request to the server through the Web page, the servlet is invoked and executed. The servlet uses the disassembly data, uses the disassembly computational algorithm, calculates the various disassembly cost components, and sends the results back to the Web page that made the request in the first place. The user may change the input to the servlet and requests the servlet execution from the server for another round of new results. This technique makes it easy for designers to investigate the “what if” scenario.

## 6. IMPLEMENTATION

The Java technology<sup>1, 9, 11</sup> has been used to implement the disassembly computational algorithm and its related client/server architecture. A servlet, written to compute the disassembly cost, has the following pseudo code:

```
Initialize the servlet
Get the components that the user selects and desires to disassemble on the Web page
Initialize disassembly cost parameters
Calculate  $TRR$ ,  $TCR$ ,  $TPC$ , and  $TDC$ 
Calculate the total benefit, total cost, net benefit, and disassembly index
Display the output on a Web page
```

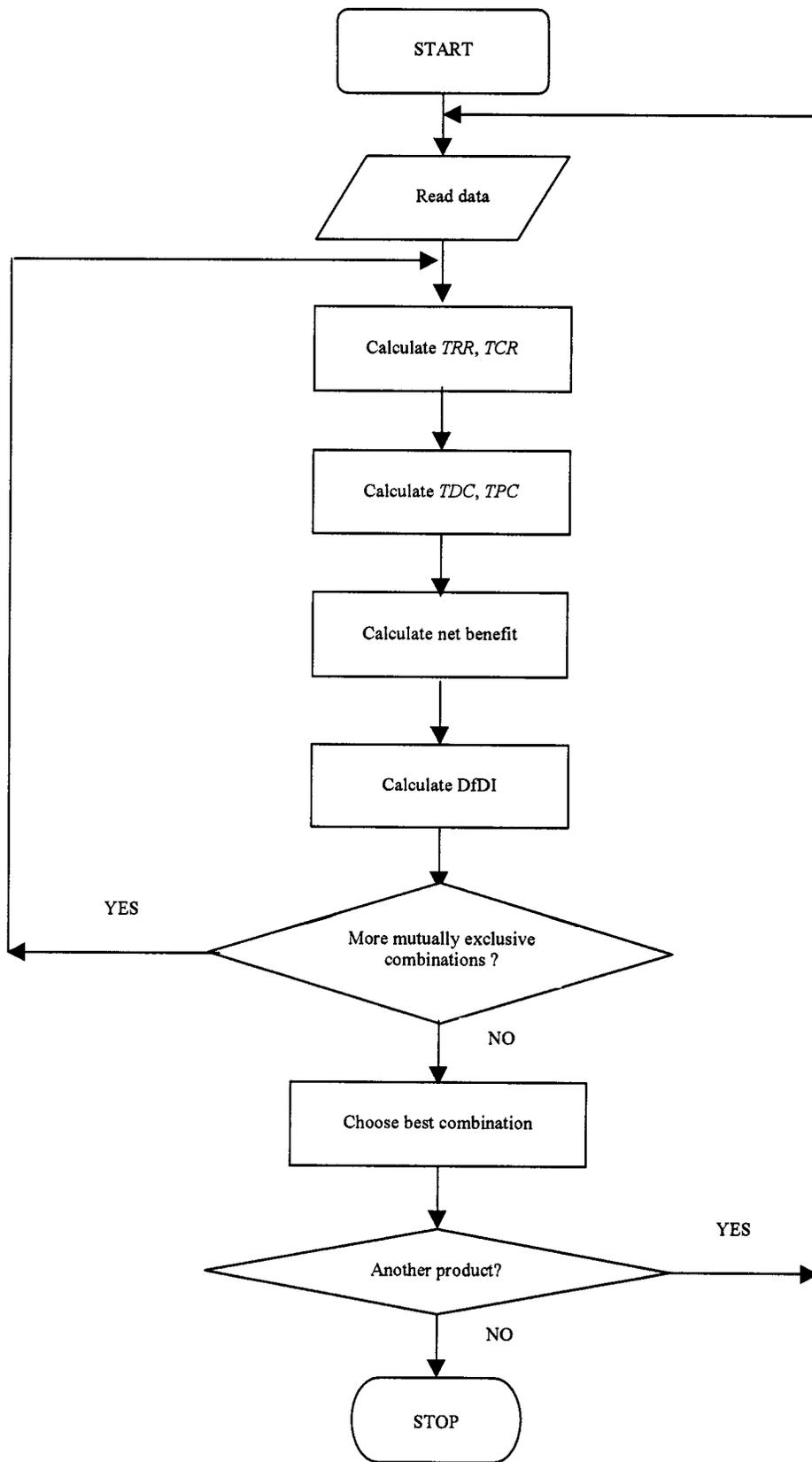


Figure 1. Disassembly algorithm

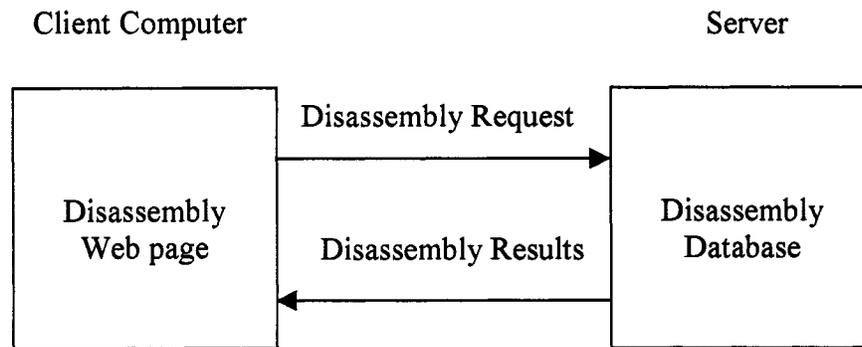


Figure 2. Two-tier client/server architecture

In addition to the servlet, a Web page has been developed that allows the designer/user to identify the components, of a given disassembly, that needs to be retrieved from the disassembly. The Web page has a form that lists all the possible product components that can be disassembled. The user can choose none, one, more than one, or all the components to be disassembled. For each case, the servlet retrieves the user's choice (selected components to be disassembled) and calculates all the disassembly costs and sends the results to the user in the form of a Web page.

## 7. APPLICATION

To test the development and implementation of the disassembly computational algorithm, we consider the disassembly of a PC (Personal Computer)<sup>16</sup>. The PC has the following six components:  $P_1$  (Housing Assembly),  $P_2$  (Power Supply),  $P_3$  (Printed Circuit Boards, PCBs),  $P_4$  (Mother Board),  $P_5$  (Floppy Disk Drive), and  $P_6$  (Hard Disk Drive). The PC tree structure is shown in figure 3. The disassembly data can be found in reference 16.

The Web page that is associated with this application is shown in figure 4. This Web page serves as the front end for the disassembly computational algorithm. The Web page allows the designer/user to choose any set of components to disassemble. The designer/user may select none, one, multiple components, or all six components to disassemble. The designer/user clicks the "Calculate Disassembly Cost" button after finishing component selection which invokes the servlet that resides on the server. The servlet retrieves the designer/user selection from the Web page, uses the selection to calculate the disassembly cost, and sends the results back to the Web page which is displayed via the client browser. The designer/use reviews the results and can repeat this process again and again, with different selected components each time. Figure 5 shows the results of the servlet.

## 8. DISCUSSION

The disassembly computational algorithm has been tested various times. Its results have been compared with manual calculations to ensure its correctness. The algorithm is general and is applicable to any product with any disassembly tree architecture. The algorithm and its developed tool should prove valuable to designers who have multiple design alternatives and they must decide which alternative is the best from a disassembly and environment point of view. The simplicity of the front end of the algorithm should entice designers to use it.

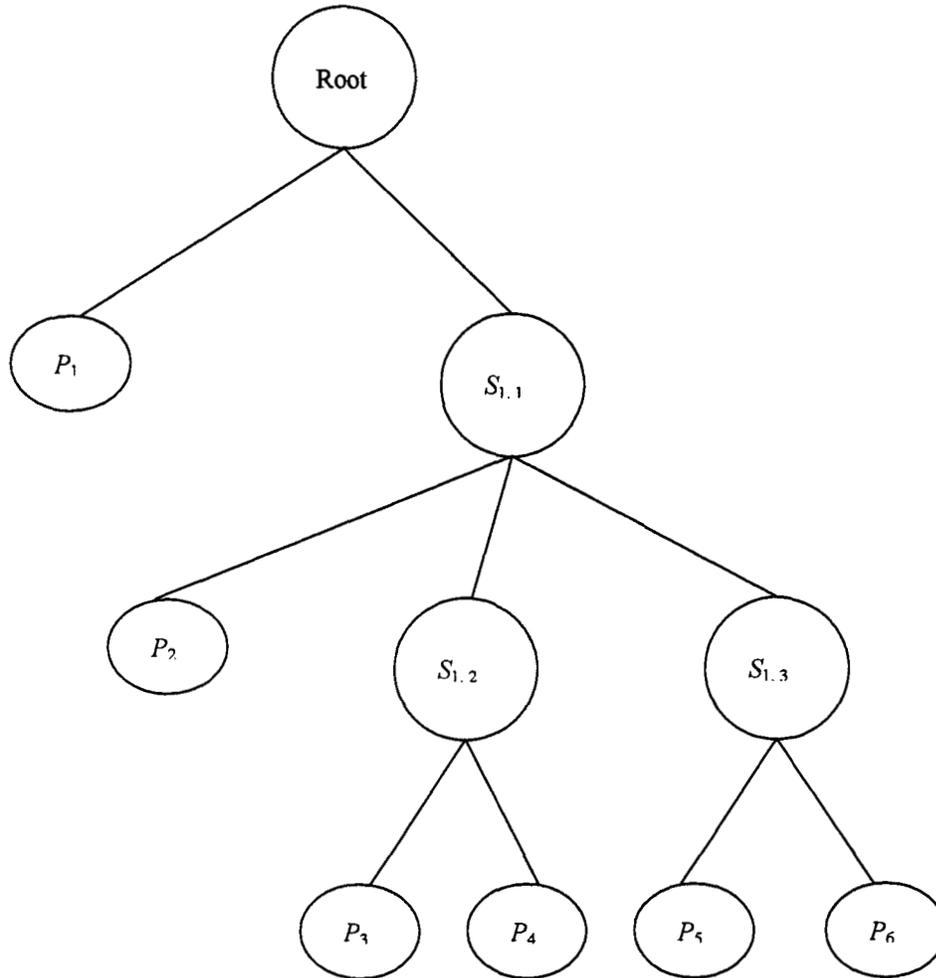


Figure 3. PC tree structure

## 9. CONCLUSIONS

Environmentally conscious design and manufacturing is becoming more crucial than in the past, in light of the dwindling natural resources and the lack of landfills to discard products at the end of their useful lives. In addition, companies have been trying to find ways to meet strict government environment regulations. Thus, we must achieve two goals. First, we need to provide designers with algorithms and tools to allow them to evaluate their designs from a disassembly point of view. Second, we need to provide managers and used markets with algorithms and tools to evaluate the profit prospects of recycling parts of products. The algorithm described in this paper achieves both goals.

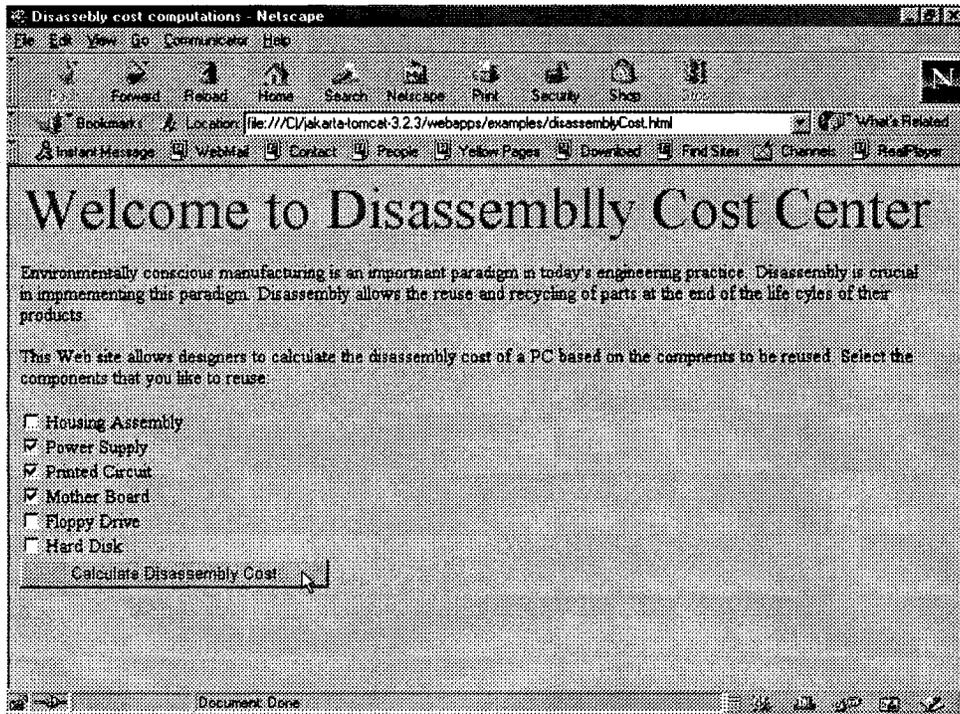


Figure 4. Web page (front end) of the disassembly computational algorithm

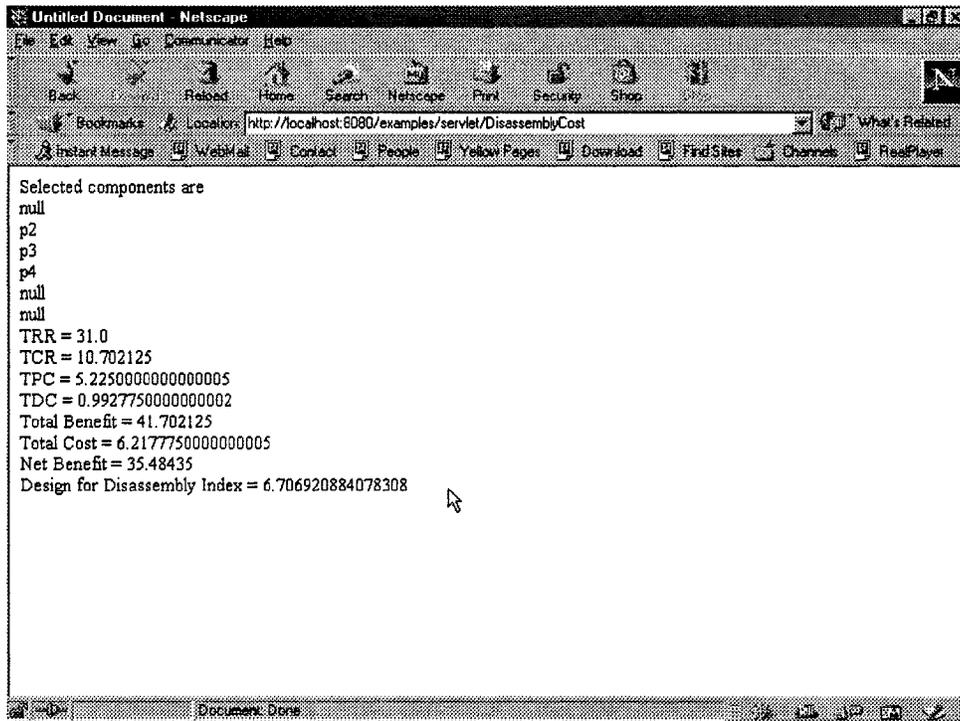


Figure 5. Results of the disassembly computational algorithm

## REFERENCES

- [1] Budd, T., *Understanding Object-Oriented Programming with Java*, Addison Wesley, 2000.
- [2] Gungor, A. and Gupta, S. M., "Issues in Environmentally Conscious Manufacturing and Product Recovery: A Survey", *Computers and Industrial Engineering*, Vol. 36, No. 4, 811-853, 1999.
- [3] Hall, M., *Core Servlets and Java Server Pages*, Prentice Hall, 2000.
- [4] Hougland, D., and Tavistock, A., *Core JSP*, Prentice Hall, 2001.
- [5] Isaacs, J. A. and Gupta, S. M., "A Decision Tool to Assess the Impact of Automobile Design on Disposal Strategies", *Journal of Industrial Ecology*, Vol. 1, No. 4, 19-33, 1997.
- [6] Ishii, K., Eubanks, C. F. and Marco, P. D., "Design for Product Retirement and Material Life-Cycle", *Materials & Design*. Vol. 15, No. 4, 225-233, 1994.
- [7] Johnson, M. R. and Wang, M. H., "Planning Product Disassembly for Material Recovery Opportunities", *International Journal of Production Research*, Vol. 33, No. 11, 3119-3142, 1995.
- [8] Jubin, H., and Friedrichs, J., *Enterprise Java Beans by Example*, Prentice Hall, 2000.
- [9] Kafura, D., *Object-Oriented Software Design & Construction with Java*, Prentice Hall, 2000.
- [10] Lee, D.-H., Kang, J.-G. and Xirouchakis, P., "Disassembly Planning and Scheduling: Review and Further Research", *Journal of Engineering Manufacture*, Vol. 215, No. B5, 695-709, 2001.
- [11] Morielli, R., *Java, Java, Java, Object-Oriented Problem Solving*, Prentice Hall, 2000.
- [12] Moyer L. K. and Gupta, S. M. "Environmental Concerns and Recycling/Disassembly Effects in the Electronics Industry", *Journal of Electronics Manufacturing*, Vol. 7, No. 1, 1-22, 1997.
- [13] Navin-Chandra, D., "The Recovery Problem in Product Design", *Journal of Engineering Design*, 5(1) (1994), 65-86.
- [14] Subramani, A. K. and Dewhurst, P., "Automatic Generation of Product Disassembly Sequence", *Annals of the CIRP*. Vol. 40, No. 1, 115-118, 1991.
- [15] Veerakamolmal, P. and Gupta, S. M., "Analysis of Design Efficiency for the Disassembly of Modular Electronic Products", *Journal of Electronics Manufacturing*, Vol. 9, No. 1, 79-95, 1999.
- [16] Veerakamolmal, P. and Gupta, S. M., "Analysis of Design Efficiency for the Disassembly of Modular Electronic Products", *Journal of Electronics Manufacturing*, Vol. 9, No. 1, 79-95, 1999.
- [17] Veerakamolmal, P. and Gupta, S. M., "Design for Disassembly, Reuse and Recycling", *Green Electronics/ Green Bottom Line: Environmentally Responsible Engineering*, Edited by Lee Goldberg, Butterworth-Heinemann; (Newnes), Chapter 5, 69-82, ISBN: 0-7506-9993-0, 2000.
- [18] Veerakamolmal, P. and Gupta, S. M., "Design of an Integrated Component Recovery System", *Proceedings of the 1998 IEEE International Symposium on Electronics and the Environment*, May 4-6, Oak Brook, Illinois, pp. 264-269, 1998.
- [19] Vujosevic, R., Raskar, T., Yetukuri, N. V., Jothishankar, M. C and Juang, S. H, "Simulation, Animation, and Analysis of Design Assembly for Maintainability Analysis", *International Journal of Production Research*, Vol. 33, No. 11, 2999-3022, 1995.
- [20] Zeid, I., *Mastering the Internet and HTML*, Prentice-Hall, 2000.