ABSTRACT

A CAD-MLCA INTERFACE FOR NEXT-GENERATION DFE TOOLS

by

Jun He

Environmental concerns and rising product disposal costs have pressed manufactures to make more environmentally friendly products and customers to use and dispose of them in an environmental responsive way. Thus the practice of Design for Environment (DFE) is becoming essential in today's industrial environment. Life Cycle Assessment (LCA) provides different ways for the environmental assessment of products. One critical step in performing lifecycle assessment of any product design is to collect design-related data. The common way based on manual input for traditional DFE tools is not only cumbersome and time-consuming but also error-prone.

This thesis presents the characteristics of a DFE tool which not only extends traditional four stages LCA model to seven stages Multi-Lifecycle Assessment (MLCA) model, but also integrates CAD tools and MLCA software. It presents an interface for the DFE tool which can automatically extract all related design information from a product's CAD database and populate to the database used for multi-lifecycle assessment. The interface design involves both database structure and population algorithm implemented in JAVA/JDBC. Thus CAD tools and multi-lifecycle assessment software are integrated to allow designers to make immediate environment performance evaluation of their product designs and thus make their changes, if needed, on-line. This work provides a robust infrastructure for the next generation DFE tools and overcomes a challenging technical barrier in the wide DFE application in industrial design practice.

A CAD-MLCA INTERFACE FOR NEXT-GENERATION DFE TOOLS

by
Jun He

A Master's Thesis
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Masters of Science in Computer Engineering

Department of Electrical and Computer Engineering

May 2001

APPROVAL PAGE

A CAD-MLCA INTERFACE FOR NEXT-GENERATION DFE TOOLS

Jun He

Dr. Mengchu Zhou, Thesis Advisor Date
Professor of Electrical and Computer Engineering Department, NJIT

Dr. Reggie J. Caudill, Thesis Co-Advisor Date
Executive Director of Multi-lifecycle Engineering Research Center, NJIT
Professor of Industrial and Manufacturing Engineering, NJIT

Dr. Yun-Qing Shi, Committee Member Date
Associate Professor of Electrical and Computer Engineering Department, NJIT

BIOGRAPHICAL SKETCH

Author: Jun He

Degree: Masters of Science in Computer Engineering

Date: May 2001

Undergraduate and Graduate Education:

- Master of Science in Computer Engineering
  New Jersey Institute of Technology, Newark, NJ, 2001

- Master of Science in Control Engineering
  Zhejiang University, Hangzhou, P.R.China, 1997

- Bachelor of Science in Mechanical Engineering
  Northwestern Polytechnical University, Xi'an, P.R.China, 1990

Major: Computer Engineering

To my beloved parents, my brother and my sister

ACKNOWLEDGMENT

I would like to express my deepest appreciation to Dr. Mengchu Zhou, who not only served as my thesis advisor, but also constantly gave me support, valuable resources and insights.

I would also like to express my sincere gratitude to Dr. Reggie J. Caudill, who not only helped as my thesis co-advisor, but also gave me encouragement and reassurance.

Special thanks to Dr. Shi for his constructive comments and recommendations and actively participating in my committee.

I would like to take this opportunity to thank Ms. Lori Nanton, Mr. Najeeb Alli, Ms. Lorietta Barnes and the research assistants at the Multi-Lifecycle Engineering Research Center (MERC) for their continuous motivation and suggestions.

Finally, I thank my friends and individuals not specifically delineated here who assisted me in this research.

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

TABLE OF CONTENTS
(Continued)

Chapter **Page**

LIST OF FIGURES

LIST OF FIGURES
(Continued)

CHAPTER 1

INTRODUCTION

1.1 Research Background

Just one generation ago, environmental quality was a little-known concept in the United States. Today we live in an age of heightened environmental awareness, fueled by public interest groups and the media. The processes involved in manufacturing and supporting most products may have adverse impacts on the environment, including the generation of waste, disruption of ecosystems, and depletion of natural resources. The current pattern of industrial development threaten to exceed the limits of sustainability in terms of resource utilization and waste management, and also poses potential threats to global climate, vegetation, and agriculture [Fiksel 96]. Therefore, the more forward-looking corporations and nations recognize that providing a suitable quality of life for the Earth's citizens involves not less industrial activity but more, not less reliance on new technologies but more, and that providing a sustainable world requires close attention of industry-environment interactions.

Both environmental concerns and rising product disposal costs have pressured both manufactures and customers for more environmentally friendly products. A large number of products related environmental laws and regulations are rapidly increasing throughout the world. Topics such as product takeback and energy efficiency are receiving a great deal of attention by the European Union and individual countries throughout the world. Monitoring, evaluating and complying with these numerous and complex laws and regulations present a major challenge for manufacturers [Thomas 95].

As a result of these economic and legislative restrictions, a firm's competitiveness in future world markets depends upon making environmental issues a central concern.

In response to the increasing pressure by organizational stakeholders, environment management is at the top of the list of environmental design objectives. There are a number of different systems-based approaches to integrating environmental issues into industry, such as Life-cycle assessment (LCA), design for environment (DFE), total quality environmental management (TQEM), ecofusion, green supply chain management, and a number of national and international environmental standards [Marion, 98]. DFE focuses primarily on a design process, whereas TQEM focuses on the management and elimination of wastes and the continuous improvement of processes and systems. Green supply chain management is linked to the external relations of manufacturing firms and involves logistic planning.

1.2 Goal and Objectives

Increasing environmental awareness in the general public concerns over the disposal of used products, and corporate efforts to buy "green" products are resulting in new challenges for manufacturers to develop cleaner technologies, environmentally friendly products and industrial production methodologies [Brinkley, 97]. Currently, to stay competitive in the marketplace, manufacturers become more aware of the environmental impacts of their products, in order to control or avoid the environmental impacts they may cause. They have undertaken environmental audits or assessments to measure the environmental performance of their products. Thus they can reduce their environmental impacts through better designs, manufacturing processes, etc. At the same time, new design technologies have contributed to lower cost, higher quality products. Different CAD software tools have provided ways to automate and simplify their most basic tasks. These CAD tools, in turn, helped engineers to devote more time for creative elements of designing, and push the limits of the available technology. The more tasks the tools can successfully tackle, the more engineers expect from their tools. However, it is not often that a single common need transcends all types of technical software users. Many issues that technical software users confront are specialized, pertaining only to a segment of the entire community [PTC, 1999].

Multi-lifecycle assessment [Caudill, 1995] is a new methodology that extends the traditional lifecycle stages set forth by SETAC and EPA [Keoleian and Menerey, 1994]. It puts greater focus on quantifying materials, energy and environmental burdens associated with end-of-life options and on value of returning parts and materials back to use, through demanufacturing, reengineering and remanufacturing. It also allocates appropriate benefits to the product over multiple generations rather than one. Multi-lifecycle engineering research aims to introduce a next generation engineering system in which the quality of the waste stream is engineered with the same concern as the product itself, and where discarded products and waste material are reengineered into valuable feedstocks.

Although many stand-alone CAD and LCA software packages are already available from different companies or independent research centers, there is the need for integration of CAD and LCA to make DFE reflect LCA requirement more efficiently and accurately. The basis of the integration of CAD and LCA software is the fact that both

LCA and CAD share many common data items which play the significant roles in LCA, such as part geometry, material and manufacturing processing.

A multi-lifecycle engineering software tool, developed by NJIT [Jin, 1999; Suratran, 2000], supports all MLCA activities. It also can be integrated into a CAD tool. Our research goal is build a critical link between CAD tools and MLCA and allow eventually the mutual exchange of design information to make DFE more effective.

1.3 Thesis Format

This thesis presents a review of the concept of Design for Environment (DFE), life-cycle assessment (LCA) and Multi-Lifecycle Assessment (MLCA) in Chapter 2. Chapter 3 presents the necessity of the integration of CAD and MLCA software. MLCA methodology focuses on multi-lifecycle of a product with the idea of disassembling an old product and passing different subassemblies and parts of the product into different ways to make them reusable. In the integrated design, CAD tools can use multi-lifecycle assessment (MLCA) as the mechanism for quantifying the impact of design and production issues. Relational database design for the MLCA software is presented in Chapter 4. In Chapter 5, two different products are used to present the implementation detail of the CAD-MLCA interface. One is TEL99 as a 'perfect' sample, and another is RVS_ELEVATOR with some errors of design information. Chapter 6 presents the contributions and limitations of this thesis and future research issues to design the next generation of DFE tools.

CHAPTER 2

DESIGN FOR ENVIRONMENT AND MULTI-LIFE CYCLE ASSESSMENT

Industry has concentrated its efforts on their production processes, with an aim to develop and modify processes to minimize environmental burdens and conserve energy and resources. The concepts of Design for Environment (DFE) and Life Cycle Assessment (LCA) are proposed to fulfill the aim [Marion, 98]. DFE is a systematic process by which firms design products and processes in an environmentally conscious way. It requires environmental considerations over a complete product life cycle in the design process. Closely linking to DFE output, Life Cycle Assessment (LCA) is a family of methods for examining and selecting materials, products, processes, and technologies of a product through every step of its life. The assessment includes the entire life cycle of the product, process or activity, encompassing extracting and processing raw materials; manufacturing, transportation, and distribution; use, reuse, maintenance; recycling; and final disposal.

2.1 Design For Environment

Design For Environment (DFE) can be defined as systematic consideration of design performance with respect to environmental, health, and safety objectives over the full product and process life cycle [Fiksel, 93]. It is a design practice that embraces all the three approaches of cleaner processes, cleaner products, and sustainable resource use. Moreover, it addresses the traditional concerns of health and safety management to the extent that they are important product design considerations. In short, DFE is the design of safe and eco-efficient products. It seeks to discover product innovations that result in reduction of pollution and waste at any or all stages of the life cycle, while satisfying other cost and performance objectives. DFE cannot be practiced in isolation - it must be balanced against other design considerations to optimize a design. In order for DFE to be integrated effectively into a new-product development process, the following key elements are required:

■ Eco-efficiency metrics, driven by fundamental customer needs or corporate goals, to support environmental performance measurement;

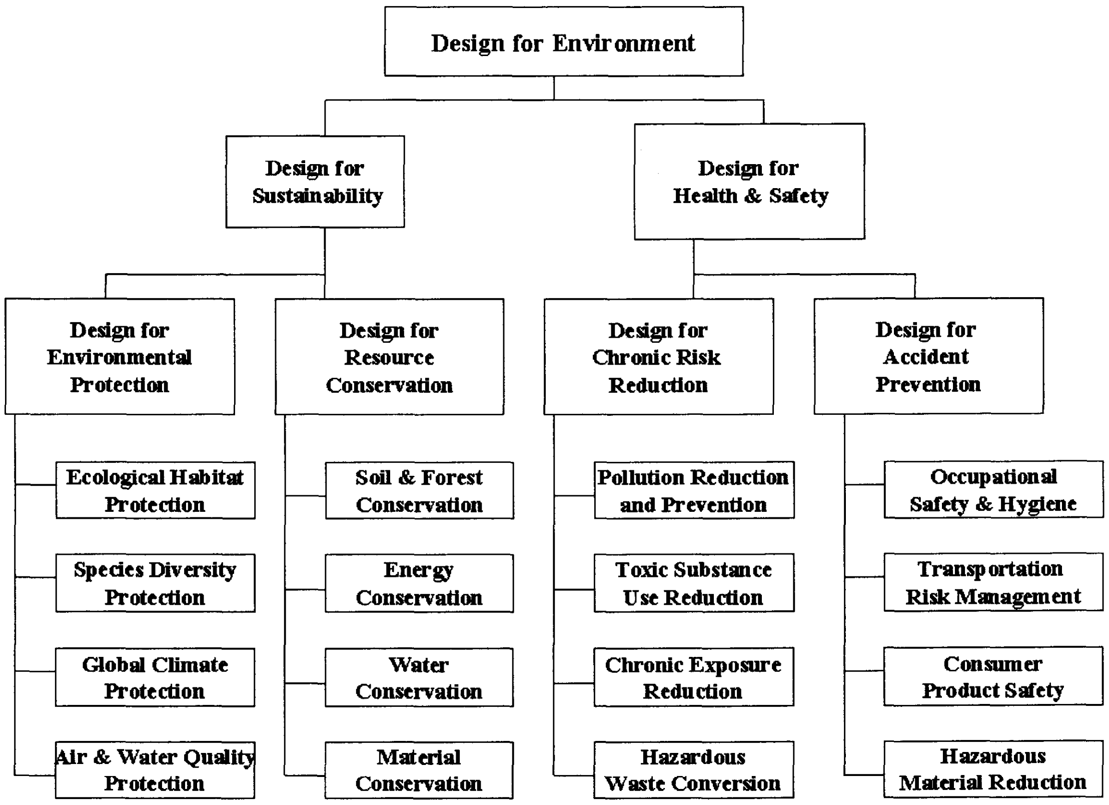

Design for Environment
Design for Sustainability
Design for Health & Safety
Design for Environmental Protection
Design for Resource Conservation
Design for Chronic Risk Reduction
Design for Accident Prevention
Ecological Habitat Protection
Species Diversity Protection
Global Climate Protection
Air & Water Quality Protection
Soil & Forest Conservation
Energy Conservation
Water Conservation
Material Conservation
Pollution Reduction and Prevention
Toxic Substance Use Reduction
Chronic Exposure Reduction
Hazardous Waste Conversion
Occupational Safety & Hygiene
Transportation Risk Management
Consumer Product Safety
Hazardous Material Reduction

Fig. 2.1 A hierarchy of DFE discipline [Fiksel, 1996]

■ Eco-efficient design practices, based on in-depth understanding of relevant technologies and supported by engineering guidelines; and

■ Eco-efficiency analysis methods to assess proposed designs with respect to the above metrics and to analyze cost and quality tradeoffs.

The practice of DFE is becoming essential in today's industrial environment, as more and more firms recognize the importance of environmental responsibility to their long-term success. DFE represents a way to achieve sustainability while seeking competitive advantage. The scope of DFE, speaking broadly, can encompass a variety of overlapping disciplines. Fig. 2.1 shows a hierarchical breakdown of DFE disciplines, most of which are routinely practiced by manufacturing firms [Fiksel, 1996].

2.2 Life-cycle Assessment

Life-cycle Assessment is a tool to evaluate the environmental consequences of a product or activity holistically, across its entire life. A complete life-cycle assessment consists of four complementary components: inventory, impact, improvement analysis, and scoping.

The inventory analysis component is a technical, data-based process of quantifying energy and raw material requirements, atmospheric emissions, waterborne emissions, solid wastes, and other releases for the entire life cycle of a product, package, process, material, or activity, which is shown schematically in Fig. 2.2. Qualitative aspects are the best captured in the impact analysis, although it could be useful during the inventory to identify these issues. In the broadest sense, inventory analysis begins with raw material extraction and continues through final product consumption and disposal. Some inventories may have restricted boundaries because of their intended use, e.g., internal industrial product formulation improvements where raw materials are identified.

Inventory analysis is considered as the only component of life-cycle analysis that is well developed. Its methodology has been evolving over a 20-year period [Graedel, 98].

The impact analysis component is technical, quantitative, and/or qualitative process to characterize and assess the effects of the resource requirements and environmental loadings (atmospheric and waterborne emissions and solid wasters) identified in the inventory stage. The analysis should address both ecological and human health impacts, resource depletion, and possibly social welfare. The key concept in the impact analysis component is that of stressors. The stressor concept links the inventory and impact analysis by associated resource consumption and releases documented in the inventory with potential impacts. Thus, a stressor is a set of conditions that may lead to an impact.

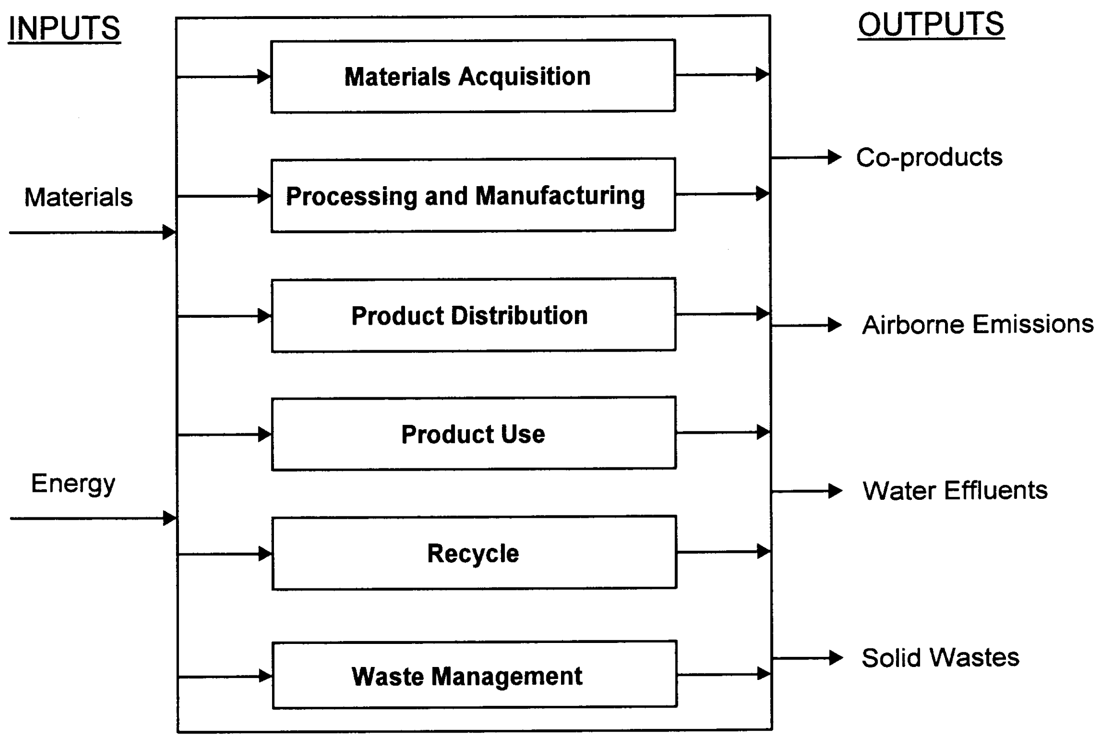


Fig. 2.2 The elements of a life cycle inventory analysis[SETAC, 1991]

An important distinction exists between life-cycle impact analysis and other types of impact analysis. Life-cycle impact analysis does not necessarily attempt to quantify any specific actual impacts associated with a product or process. Instead, it seeks to establish a linkage between the product or process life cycle and potential impacts. The principal methodological issue is managing the increased complexity as the stressor-impact sequence is extended.

The improvement analysis is a systemic evaluation of the needs and opportunities to reduce the environmental burden associated with energy and raw material use and waste emissions throughout the life cycle of a product, process, or activity. This analysis may include both quantitative and qualitative measures of improvements.

Scoping is one of the first activities in any life-cycle assessment. During scoping, the product, process, or activity is defined for the context in which the assessment is being made. The scoping process links the goal of the analysis with the extent, or scope, of the study, i.e., what will or will not be included [Keoleian et al, 94].

The essence of life-cycle assessment is the examination, identification, and evaluation of the relevant environmental implications of a material, process, product, or system across its life span from creation to waste or, preferably, to re-creation in the same or another useful form [Graedel, 98].

2.3 Multi-Lifecycle Assessment

Lifecycle assessment methodology is the traditional framework set up jointly by U.S. Environment Protection Agency (EPA) and Society for Environmental Toxicology and Chemistry (SETAC). It considers all the lifecycle stages of a product, process, or activity from raw material extraction to final disposal. Multi-Lifecycle Engineering Research Center (MERC) of NJIT has extended the structure of traditional LCA to include explicit consideration of demanufacturing, remanufacturing, reengineering and reuse-extending LCA's to the realm of multi-lifecycle engineering. These end-of-life recovery processes have been modeled to account for material flows, energy usage and environmental burdens associated with recovery and reprocessing of components and basic materials [Caudill, 95].

Multi-Lifecycle Assessment (MLCA) is a new approach in today's environmental area [Caudill, 97]. It is based on the principle of sustainable economy where competitiveness is achieved with environmental responsibility. MLCA takes a systems perspective and considers fully the potential of recovering and reengineering materials and components from one product to create another, not just once, but many times. Multi-Lifecycle assessment (MLCA) also address these lifecycle stages beginning from raw material extraction to final disposal, but differs from tradition LCA in the last stage and in the overall perspective of thinking about the LCA methodology.

The term multi-lifecycle emphasizes the fact that technology developed here would enhance the use of discarded scrapped products, or other waste-streams so that materials and components can be used over more than a single product lifecycle. In this respect MLCA does not differ quantitatively in terms of inventory analysis as compared to traditional LCA, but tries to develop a systems perspective towards this area. It looks backwards in time to designing products such that the materials used in them have multiple lives and thus uses after their first useful life. This requires a clear vision and understanding of the product from its raw material extraction till its usage.

And hence efficient demanufacturing of a product is one of the prime goals of multi-lifecycle engineering. Design for disassembly helps in attaining this goal and efforts are being made towards developing a methodology for it.

Multi-Lifecycle assessment can be described as a cradle-to-grave-to-cradle analysis while traditional LCA is a cradle-to-grave analysis. MERC has rebuilt the traditional four stages lifecycle into seven stages multi-lifecycle. These stages are Material Extraction, Material Synthesis, Manufacturing and Assembly, Packaging and Distribution, Demanufaturing, Remanufacturing, and Reengineering. The details of each stage are described below.

2.3.1 Material Extraction and Synthesis Stage

From about 90 chemical elements, some of which can't be efficiently recovered, scientist and technologist have been able to refine, combine, and take apart and recombine elements to produce thousands of metals, wood products, plastics and ceramics that are chemically different and have distinctly different characteristics. Material's shapes and forms have provided the basis from which an almost infinite number of manufactured goods have been developed.

Material production has been divided into two separate stages, materials extraction and materials synthesis. This breakdown is useful in defining the depth and level of the study as it gives more options for designers to evaluate their product based on specific needs. Material Extraction consists of all the processes that are required to extract material from the earth. Materials that can be found and used in their natural form usually require some type of primary processing. For example, sand is washed and graded; trees are cut into boards and planks; and coal is washed, cleaned and sized before it is shipped to the customer. Materials that are combinations of natural materials, such as iron or steel, may require several different processes during the primary stage in order to get them ready for use. Other materials, such as plastics, may be by-products (additional use) of natural materials that are intended for different uses, such as oil for energy [Wright and Helsel, 96].

Material Synthesis consists of intermediate processing, final processing and end-use preparation. Most plastics are formed from natural materials, which are carbon based. They are developed through chemical and mechanical processes. Heat, pressure, and chemical actions are used to separate the atoms and molecules of the raw materials, which are recombined to form new materials. Through the use of mass and energy balance equations, the total energy, mass and environmental requirements for the production of each material are determined.

2.3.2 Manufacturing and Assembly Stage

This stage includes processing of parts, subassemblies and finalsubassemblies. Finalsubassemblies are the assemblies that are not disassembled during the demanufacturing process. For example in a *computer*, *motor* can be a finalsubassembly that is not disassembled during the disassembly of the computer. The initial inputs to a part production process are reused parts that are recovered in the demanufacturing process and raw/virgin, recycled and reengineered material from other (or same) products. While the inputs to the subassembly production process includes refurbished subassemblies that are also recovered from demanufacturing and feedstock inputs rather than materials. The outputs from these two processes are integrated into a final assembly process, which result in the finished product [Badwe, 97].

2.3.3 Packaging and Distribution Stage

Another stage that is further quantified in MLCA methodology is the packaging and distribution stage, which has been separated from the production phase as a unique stage. This stage quantifies the materials used in the packaging process, the packaging process itself, as well as methods of transportation, distance traveled and energy and emissions associated with these processes.

2.3.4 Use Stage

Measuring the environmental performance of a product during the use stage is very crucial, since energy consumption during this stage is usually the maximum. Generally, electronic products consume energy during use in four modes: Active, Idle, Power Save, and Off modes. Energy consumption must be measured in each of these modes, in order to quantify energy consumed during the product lifecycle. Similarly, environmental burdens are measured during those modes and quantified. After the product is consumed and disposed of, it is sent to demanufacturing, where its end fate is identified [Jin, 99].

2.3.5 Demanufacturing Stage

If during the product design stage, designers do not concern with the disassembly or the reuse of parts, then disassembly may be an expensive process and is usually not a viable end of life option. As a result, products having good recycling value are disposed of or recycled in their entirety. The latter leads to a lower material recycling value. For example in the case of computers and copiers, parts such as CRT glass, keyboard caps, drive motors, transformers, and steel casing, could all be given a new life if they could efficiently and economically be disassembled [Das et al., 98].

Disassembly is the process of physically separating parts and subassemblies in a product. To recover the maximum value from the discarded products, product disassembly stage needs to be carefully analyzed in terms of sequence dependency between components and the required level of disassembly [Das et al., 99].

The process of disassembly involves several activities, including collection of parts, sorting of parts, product/part handling, disassembly instruction, part separation, part cleaning, part testing and inspection. These activities can further classified into

1. Unfastening Processes – where a fastening devise is removed in an operation, which reverses the assembly fastening action; and
2. Disassembly Processes – All other activities that facilitate the separation of a product into its parts.

2.3.6 Reengineering Stage

The main option to consider and where MLCA differs from LCA is in the recovery of material and new life options of a product. LCA addresses two types of recycling processes, open loop recycling and closed loop recycling. In an open-loop recycling system, a product made from virgin material is recycled into another product that is not recycled, but disposed, possibly after a long-term diversion, thus giving only one life to the product. Whereas closed-loop recycling occurs when a product is recycled into a product that can be recycled over and over again. LCA looks into this as two separate distinct recycling options. This is one area where MLCA plays its role, in addressing these two recycling options simultaneously, rather than in isolation, at the end of a product's life as well as throughout its life [Jin, 99].

MLCA tries not to landfill any material if possible. It finds new options for the waste disposed at every stage so that it can be reengineered into useful products and not just once but again and again. This reengineering stage acts as a link that closes the lifecycle loop. Reengineering involves characterization of waste streams and the reformulation of materials derived from waste streams. Five major reengineering processes were identified for recovered materials: Reprocess, Compatibilize, Pyrolysis to Fuels, Pyrolysis/Hydrolysis to monomers, and shredding of metals. The major material inputs to this stage are obtained from demanufactured products, through process such as shredding and separation [Al-Okush, 99].

2.3.7 Remanufacturing Stage

The remanufacturing stage is where the parts and subassemblies are refurbished. These refurbished products could then be used in new products at the production stage or for replacements and maintenance at the use stage or could be sent for demanufacturing. Parts and subassemblies entering the remanufacturing process can either be from demanufactured products, or from the manufacturing process. The flow of remanufacturing process is as follows:

Sort $\longrightarrow$ Pretest $\longrightarrow$ Disassemble $\longrightarrow$ Clean $\longrightarrow$ Rebulid/ Upgrade $\longrightarrow$ Reassemble $\longrightarrow$ Final Test $\longrightarrow$ Packaging $\longrightarrow$ Transport

Energy and environmental burdens must be quantified at each of the above steps. The result is either an assembly or a product that is fed back into the manufacturing process or back to the use stage through maintenance. Also, products and subassemblies that can't be remanufactured are sent to demanufacturing for further processing. Fig. 2.3 shows an overall lifecycle data flow for MLCA analysis and modeling.

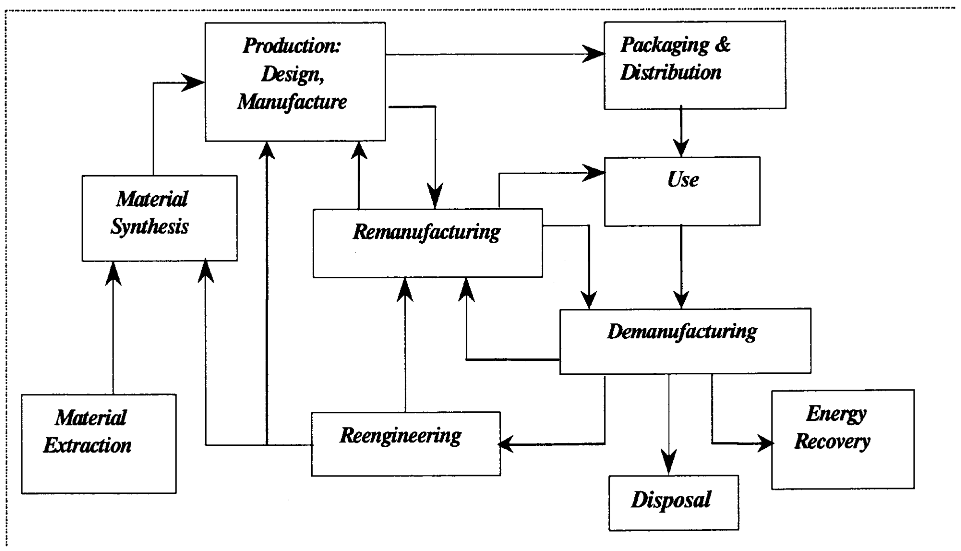


Fig. 2.3 Lifecycle Data Flow for MLCA [Jin, 1999]

CHAPTER 3

INTEGRATION OF CAD AND MLCA SOFTWARE

3.1 LCA Software

The LCA process encompasses the identification and quantification of energy and material usage, as well as environmental releases across all stages of a life cycle; the assessment of the impact of these energy and material uses and releases to the environment; and the evaluation and implementation of opportunities to effect environmental improvement [Fava, et. al., 92].

In the recent years, LCA has gained general acceptance as a tool with a range of uses, such as environmental labeling, product environmental improvement, eco-design, and policy evaluation. As the acceptance of LCA has increased, so has the development of software tools and databases for performing LCA [Menke, Davis and Vigon 96].

3.1.1 General Structure of LCA Software

Stand-alone LCA software has evolved rapidly over the past several years from largely spreadsheet-based systems to graphically oriented systems employing standard Windows features and appearance. The most common type of LCA software model, as is distinguished from design or engineering LCA modeling tools, is referred to as an *input-output model* [Curran 96]. A typical structure of stand-alone LCA software is shown in Fig. 3.1, which consists of a user interface, database, computational engine, and report processor. At first, product data and information are stored into LCA databases through user interfaces, then, data processing is carried out with the computational engine, and calculation results are treated in report processor to give a direct and clear reports to a user.

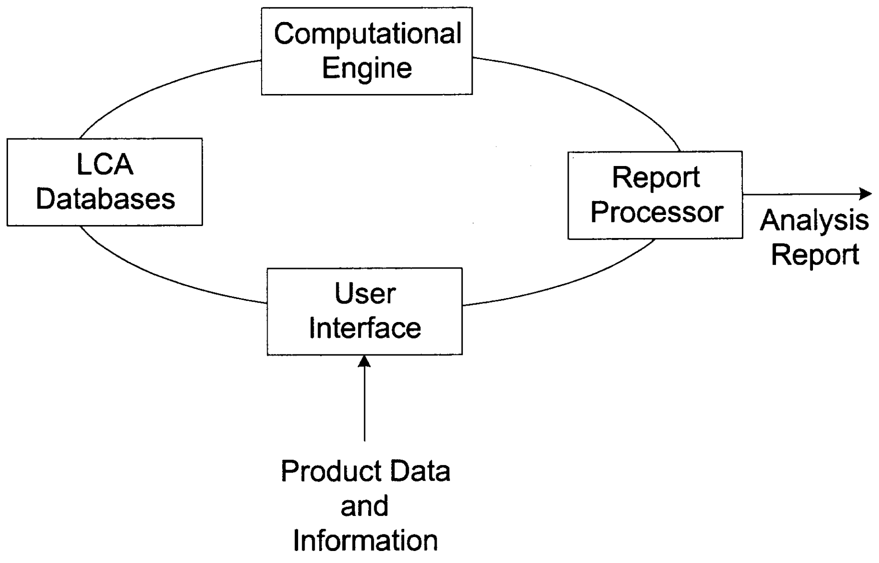
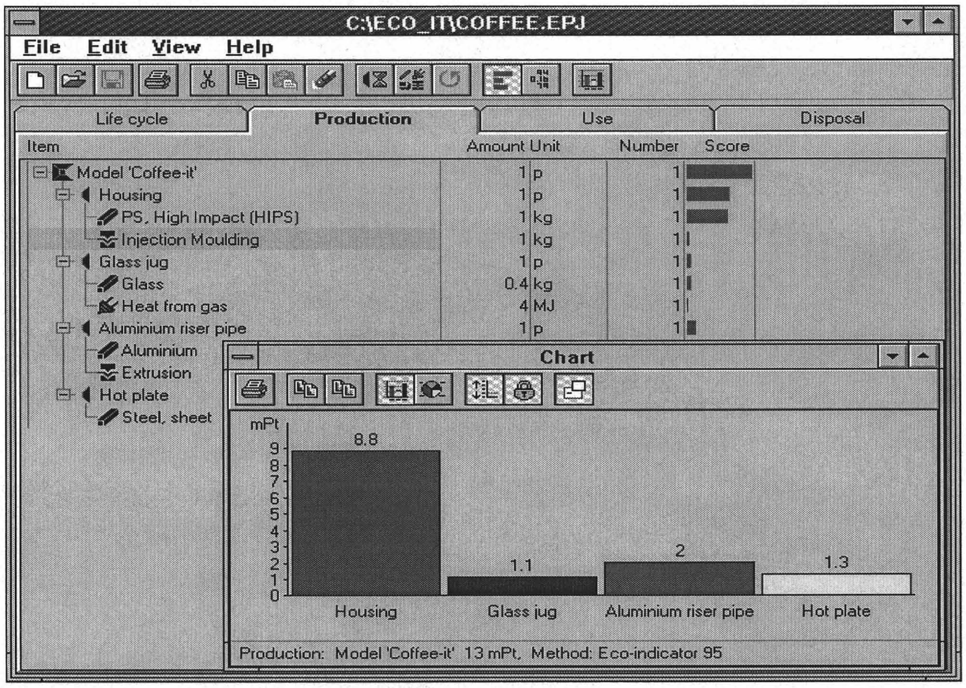


Fig. 3.1 Typical Structure of Integrated LCA Software

Targeted users of LCA software are expert LCA practitioner and/or general users. It provides recommendations on materials and process choices based on life-cycle considerations. Some software also has the ability to select alternative processes for manufacture of an item. Within the software, a database and an expert system have been incorporated to translate a designer's choices into the necessary inventory and impact assessment computations. Choices on the depth and breadth of LCA may have been pre-selected by a developer in order to balance the complexity with the multidimensional nature of a decision process.

3.1.2 LCA Tools

Many LCA tools have been available for implementing a full LCA analysis. Since it is unlikely that one single LCA methodology can be optimal for all LCA analysis, differences can be found in these tools, depending on the boundaries set by the tool and the specific problems it is designed to solve. Some software deals with energy at just one life-cycle stage rather than the total lifecycle. Some varies in the type of databases of materials it uses.



Item	Amount	Unit	Number
Model 'Coffee-it'	1	p	1
Housing	1	p	1
PS, High Impact (HIPS)	1	kg	1
Injection Moulding	1	kg	1
Glass jug	1	p	1
Glass	0.4	kg	1
Heat from gas	4	MJ	1
Aluminium riser pipe	1	p	1
Aluminium			
Extrusion			
Hot plate			
Steel, sheet			

Production: Model 'Coffee-it' 13 mPt, Method: Eco-indicator 95

Fig. 3.2 Eco-it software [Eco-it, 2001]

■ Eco-it

Eco-it shown in Fig. 3.2 is a LCA tool developed by PRé Consultants, which helps designers measure and optimize the environmental performance of a product in its design phase. It specifies a product with its life cycle by inputting the material and process information of that product, calculating its environmental load, and showing the product contribution result for environment.

■ TEAM

TEAM is a professional LCA software program tool for Environmental Analysis and Management [TEAM]. It allows users to build and use a large database and to model any system representing the operations associated with products, processes and activities It is designed to describe and model complex industrial systems and to calculate the associated Life cycle inventories, Life cycle potential environmental impacts, and Process-oriented life cycle costs.

■ SimaPro 4.0 Software

SimaPro, shown in Fig. 3.3, is the most widely used tool for the environmental assessment of products which is developed by the product ecology consultants in Netherlands [SimaPro]. This software allows users to utilize LCA data to analyze the environmental impact of their products. It provides an extensive database of materials, processes, energy sources, transportation, use, and waste treatment scenarios. The software has the ability to conduct an inventory analysis and an impact analysis on the product. Users must first describe a product, specifying its various parts, subassemblies and components. To view the structure of the product, SimaPro provides a process tree that displays the process and materials used to create the assembly.

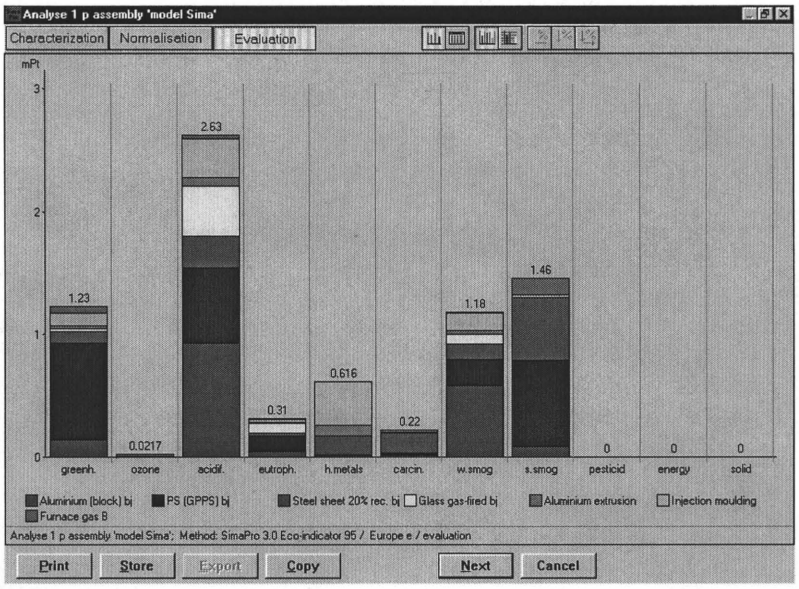


Fig. 3.3 Evaluation graph for model Sima [SimaPro, 2001]

3.2 Evaluation of LCA software

3.2.1 Computer Requirements and Interface

Computer requirements are the basic hardware requirements for every LCA software tools. Memory requirements and minimal processing unit capabilities are the primary Hardware Requirements. Software Requirements are operating systems and databases that is required by the LCA to operate properly. In other words, they may include the type of platform (Macintosh, Windows, and Unix), as well as supporting applications not supplied by the LCA tools (e.g., spreadsheet applications such as Excel).

An interface describes the basic screens with which a user must interact while developing and manipulating the product/service life cycle under investigation. This interface, and the development of system life cycles is further evaluated and described in the following section.

3.2.2 System Definition

System definition includes the three evaluation criteria of

- System development ;
- System editing; and
- Archiving.

System development describes how a user can specify steps within a manufacturing process or stages within a product life cycle to the system under investigation. This includes how flows of materials, emissions, and other burdens are specified within each step/stage, and how transportation and energy requirements are incorporated into the system. The different ways in which each software tool defines functional flows/functional units are also included under this heading.

System editing is a brief explanation of system editing capabilities and limitations as a user develops a new or changes an existing life-cycle system. Adding or deleting steps/stages, changing flows, and manipulating the developed system within the software interface are considered in this section. Data editing is addressed separately in the general category Data and Data Management.

Archiving as an evaluation criterion assesses the capability of each LCA software tool to reuse previously defined systems (or sections of systems) in new life-cycle evaluations. As a library of life-cycle systems is developed, a user may find it necessary and convenient to reuse all or some of the saved information. For example, a common energy matrix or waste disposal scenario may be used for many different life-cycle systems.

3.2.3 Data and data management

There are a number of issues surrounding life-cycle data, database, and data management capabilities which must be addressed when assessing the capabilities of each software tool. Under the criterion of Data and Data Management there are six areas of interest: embedded data; data quality indicators; other descriptive fields; data protection; data editing; and user-defined data.

Embedded Data describes the types of data available within databases accompanying each software tool. A brief assessment of data quality is also included under this heading.

The various ways in which a user can specify data quality indicators are included under Data Quality Indicators. Text fields which allow a user to specify the source of data, data collection, geographic regions, etc., are features addressed within this criterion. The quality of embedded data is not assessed under this heading. Other descriptive text fields, such as system title, process notes, etc., are included under the Other Descriptive Fields. User-defined descriptive fields are features which strengthen the life-cycle assessment process, as well as simplify the interpretation of the assessment results.

Data Protection and Data Editing document the various ways in which the information contained in the database (whether embedded or user-defined) is presented to users, shielded and /or protected from other users, and available for editing. Data protection considers both embedded protection and user-defined data protection. The protection of embedded data can include complete inaccessibility to the data, view only, or copying/editing capabilities. The protection of embedded data can include the use of user names and passwords, levels of security clearance, etc. User-defined data protection can include, for example, features which offer data access and editing capabilities to only those users who created the data set, as well as various levels of access similar to those described for embedded data protection.

3.2.4 Flexibility

Three separate criteria were identified under the general heading of system flexibility: unit flexibility, use of formulas, and burden allocation. Unit Flexibility describes whether a tool supports user-defined units or whether a user must convert all entries to consistent software-defined units. The Use of Formulas offers another degree of flexibility. To determine/specify material flows, energy requirements and environmental releases based on used-defined variables can permit a user to develop a more dynamic system. Allocation of burdens among co-products and/or open-loop flows is an issue of interest for all LCA practitioners. There are various ways by which burdens are allocated to a product or service, e.g., by weight and economic value.

3.2.5 Calculation and Comparison

Uncertainty analysis, impact assessment, and comparison of results represent three data manipulation capabilities which may or may not be a function of each LCA software tool. Calculation and Comparison, as an evaluation criterion, assesses each tool for these manipulation capabilities.

With each bit of information and data entry within an LCA, there exists a degree of uncertainty. The capability of an LCA software tool to manage this uncertainty may be a characteristic of importance to a user. Therefore, the various methods to perform uncertainty analysis, such as sensitivity analysis, within each software tool are included under Uncertainty Analysis.

Impact assessment includes the classification, characterization, and valuation of life-cycle inventory results. The flexibility to incorporate user-defined parameters for these and other assessment methods is of primary interest to many LCA practitioners and users of LCA software tools.

Comparison of Results compares the results (inventory or impact assessment) of two or more systems. For example, a comparison may be of two identical systems with different recycle rates or raw material inputs. It may be of two completely different and competing products/technologies to accomplish the same function. The results may includes airborne emissions, water effluents, solid wastes, energy consumption, and environmental cost.

3.2.6 Outputs and Exports

Outputs and Exports, the final assessment category, evaluates the various ways in which a life-cycle system and calculated results can be viewed, printed, exported, and manipulated in other ways.

The ability to utilize the information created in the LCA software tool in other computer applications for report purposes, presentations, and further manipulations is yet another capability of each software tool. Export capabilities such as data export, inventory export, and impact assessment export are among the factors included in this criterion [Menke, Davis and Vigon, 96].

3.3 Integration of CAD and MLCA software

3.3.1 The need for integration of CAD and MLCA software

One of the barriers to concurrent engineering, especially in high-technology industries, is the reliance of engineers on design and manufacturing automation tools that do not allow cross-functional integration. The environmental consideration needs LCA to provide DFE support during a product design and development process. There is a growing body of literature devoted to the development of DFE tools using lifecycle assessment (LCA) as a mechanism for quantifying the impact of design and production issues over the working life of a product. However, as shown in Fig. 3.4, LCA is still a relatively separate part from design tools.

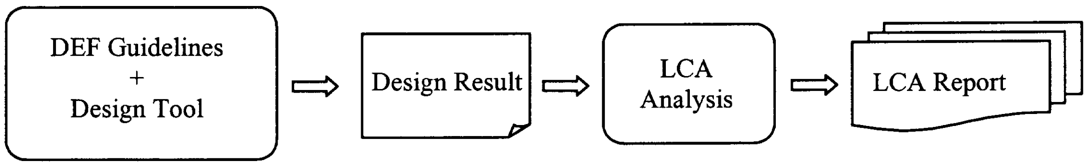


Fig. 3.4 Data flow from designs tools to LCA

Many designers realize that the DFE will be more eco-efficient if LCA tool can be completely integrated with CAD software. Furthermore, design challenges posed by a multi-lifecycle strategy are significantly more complex than traditional product design. Designers must look forward in time to maximize the product's end-of-life yield of assemblies, parts and materials while looking backward to the world of existing products for feedstock sources for the current design. Thus, a CAD-MLCA interface must be designed to allow CAD tools using MLCA as a mechanism for quantifying not only the impact of design and production issues over the working life of a product but also the demanufacturing and reuse issues.

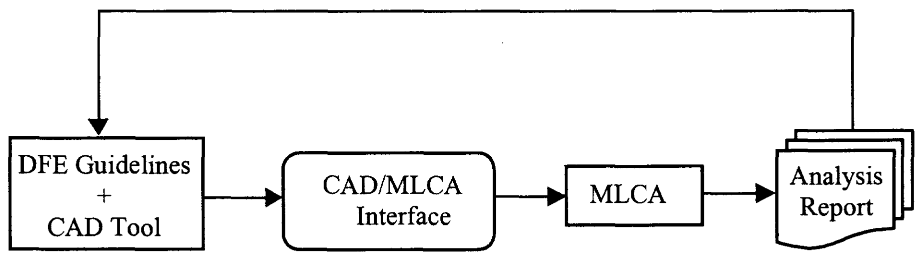


Fig. 3.5 Integration of CAD tools and MLCA into an integrated DFE tool

To consider the environmental impact for all stages of a lifecycle at the design stage requires bi-directional flow of information between design software and MLCA software. CAD software contains the product level description. The MLCA software requires this product level information to perform the analysis of life cycle stages. The flow of information from CAD software to MLCA software is facilitated by the CAD-MLCA interface which will provide the MLCA software the design information from CAD database. After performing the MLCA analysis, the designer of a product receives environmental performance characteristics leading to product modifications which may make that product more eco-efficient and environmentally friendly. Thus designers can make changes to a product in the CAD software and again the updated information about modified one can be supplied to the MLCA software using the CAD-MLCA interface. This flow of information between CAD tools and MLCA is illustrated in Fig. 3.5. Depending on different requirement, CAD tools may contain DF*X*, where X can be assembly, manufacturablility, and environment. In the design of integrated DFE software package, this kind of closed-loop development model is used, which makes DFE more accurate and efficient.

It is interesting to look inside the integrated DFE software package and compare two individual databases of CAD and MLCA. DFE must consider not only short-term environmental goals, but also how to achieve the longer-term vision of a sustainable society. In order to achieve sustainable development, it is necessary to reduce the consumption of resources and energy. CAD tools used for designing a product do not include demanufacturing, remanufacturing or reengineering data that MLCA includes. Thus MLCA takes into account the factors for resources and energy considering demanufacturing, remanufacturing and reengineering processes. The CAD database and MLCA database can be represented as shown in Fig. 3.6. The CAD/MLCA shared database will provide not only to designers but also for demanufacturers and MLCA analysts to play an active role in achieving this vision of sustainability.

A longer-term strategy for the design of products or services is necessary to move beyond current incremental environmental improvements. The CAD/MLCA shared database should contain the data required for calculating environmental burden and data required for giving intelligent design suggestions such as material selection, manufacturing process or disassembly methods.

A problem associated with MLCA software is that users of MLCA have to provide a significant amount of information about a product for which the environmental burdens are being analyzed. The information required is assembly and subassembly names, parts, materials, manufacturing processes, and the structural relationships between assemblies and their parts.

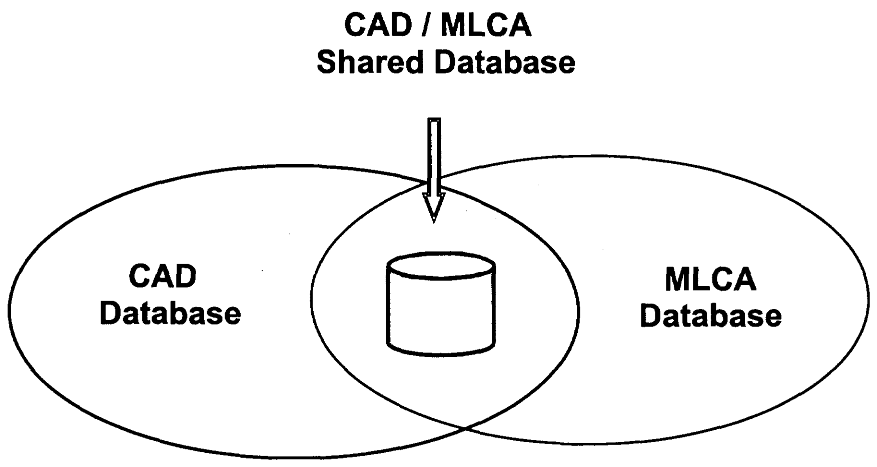


Fig. 3.6 CAD and MLCA database with shared part

At present, users have to enter most information manually. Then hundreds or thousands of entries, depending on the assembly and design complexity, will need to be entered. Entering data is a frustrating and error prone activity for users. In Fig. 3.6, the design information used in MLCA software is actually the shared part of CAD and MLCA database. That means the database has already existed after a product is designed. Unfortunately, the database about design information is normally stored with a specified format which can be recognized only by design software itself. To avoid the manual work and associative problems, the CAD-MLCA interface has been designed as a utility part yet critical link for the integrated DFE package to extract data from CAD database and convert it to a standard format which can be used by general purpose programming language. Then the data can be populated to MLCA database automatically. Thus, it dramatically improves the overall performance of the integrated DFE software package and makes it more applicable for industry.

3.3.2 Interface Requirement

As MLCA software is data centric of the integrated DFE package, it requires a significant amount of data from users. If they have to manually populate all the data items for the database, MLCA software is cumbersome to them and potentially error-prone. As product description data plays an important role in MLCA analysis, the main task of a CAD-MLCA interface is to populate the data from CAD package into the MLCA database programmatically. Thus, this interface has a significant contribution to make MLCA more user friendly and less error prone by automatically transferring data from CAD software to the integrated database.

Normally, the major users of MLCA database are the MLCA analyst, demanufacturer and product/process designer. Users require a clear understanding of product description for the analysis. If other users such as demanufacturers must enter the same product information without knowing the correct information, the analysis may be misleading. As long as the interface imports data directly from CAD software, correct information specified by designers is available to MLCA analysts and demanufacturers. Consequently, all can perform their analysis with the least required effort. Thus CAD-MLCA interface serves the information sharing between different applications/users. A CAD-MLCA interface has following characteristics:

- Reduce tedious manual input and thereby reduce potential for errors in data,
- Provide consistency in data and avoid redundancy,
- Make data population fast, correct and broadly available, and
- Allow version control of product design.

3.3.3 Product Tree Structure and Component Extraction Algorithm

There are different modeling methodologies to represent different MLCA stages [Zussman and Zhou 2000; Tang, Zhou and Caudill 2000]. The integrated DFE software package employs a tree structure to represent a product. The root of the tree, as shown in Fig. 3.7, is the product itself. The assembly at level k can be dissembled into several subassemblies and parts at level k+1. A part which cannot be separated or is no longer worthy to be disassembled is a leaf of the tree. A recursive algorithm to extract data items from the database is as follows:

```
Algorithm    Procedure EXTRACTION (n: node);
             Begin
                extract data item
                if  node type is assembly
                   for each child  c of n, if any, in order from the left do
                       EXTRACTION (c)
             end
```

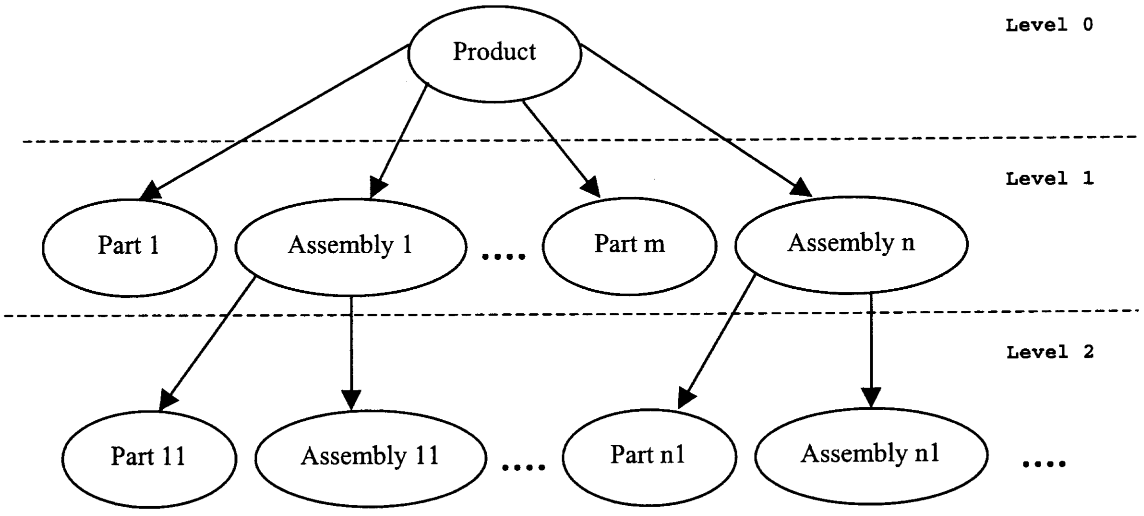


Fig. 3.7 Tree Structure of Product

3.3.4 Database Population for MLCA

Many data items in product CAD files are related to different stages of MLCA. However, not all data plays the equal roles. Some data has greater impact on the outcome of MLCA than others. The significance of each data item to the MLCA result can't be determined when we are still at product design stage. The test of whether a CAD data item is significant to MLCA is the sensitivity of the final analysis result to the data inclusion, exclusion or change. The typical way is conducted by varying input data and recalculating the MLCA output to see the result. As many data items as possible should picked up from CAD output and populate to MLCA. On the other hand, depending on its own characteristic, different MLCA software needs different level of data. The choice of data item set of different CAD-MLCA pair is variable and extensible. However, the following product information is essential to any MLCA database.

■ Product Structure

To understand its characteristic, we should generate the structure of a product with all its separate components. It is straightforward to describe the product structure as a tree structure with the product itself at the root and separate parts as tree leaves. Each component holds a fixed position in this product tree. The information necessary to create the tree structure is component process depth (level), parent-child relationship, and path identification.

■ Component Type

Basically, there are two different types of components: part and assembly. MLCA uses different algorithms and methodologies to process different types of product components corresponding to each MLCA stage. For example, if the component is an assembly, the algorithm traverses all its children. If the component is a part, the algorithm goes one level up to its parent after extracting its information.

■ Material

Material selection is a fundamental part of design. It offers many opportunities for reducing environmental impacts. Both raw material and recycle material are considered. Then the environmental impacts caused by their material acquisition, processing, use, and retirement are estimated by MLCA.

■ Geometry Parameter

In MLCA, every product manufactured has a packaging, distribution and transportation stage in its life cycle. The most common attribute for distribution and transportation is that it involves a change in location or physical configuration of a product. Therefore, every component's geometry parameters, e.g., surface area and volume, are very important information for these stages. Surface area will also be used to estimate the environmental impact if surface coating and painting processes are involved in the manufacturing and remanufacturing processes.

■ Unit

In some cases, especially in the cases of huge and complex products, different components of the same product can be designed by different designers and companies in different countries. Although international standards do exist, different designers, especially from different countries, may adopt different units. Most of CAD packages allow components with different units to be assembled to form a product. Obviously, any geometry parameter without definite unit will be meaningless to MLCA.

CHAPTER 4

DATABASE DESIGN FOR MLCA SOFTWARE

Basically, MLCA software can be viewed as a database system (DBS) which consists of software, the database management system (DBMS) and one or several databases. A DBMS functions as an interface between the users and database. It ensures that users adequate and efficient access to the data, and that the itself is resistant to hardware and software failures and can be persistently stored over long periods of time independent of the programs that access it.

4.1 DBMS Architecture

A Database Management Systems (DBMS) is a collection of programs that enable users to create and maintain a database. The DBMS is hence a general-purpose software system that facilitates the process of defining, constructing, and updating database for various applications. Defining a database involves specifying the data types, structures, and constraints for the data to be stored in the database. Constructing a database is the process of storing the data itself on some storage medium that is controlled by the DBMS. Manipulating a database includes functions such as querying the database to retrieve specific data, updating the database needed to reflect changes, and generating reports from the data [Elmasri, 94].

A widely adopted architecture for database systems is the three-schema architecture. The goal of this three-schema architecture, illustrated in Fig. 4.1, is to separate the user applications and the physical database. It has three important characteristics:

a) Insulation of programs and data.

b) Support for multiple user views

c) Use of a catalog to store database description (schema).

In the three-schema architecture schemas can be defined at the following levels:

• The internal level has an internal schema, which describes the physical storage structure of database. The internal schema uses a physical data model and describes the complete detail data storage and access paths for the database.

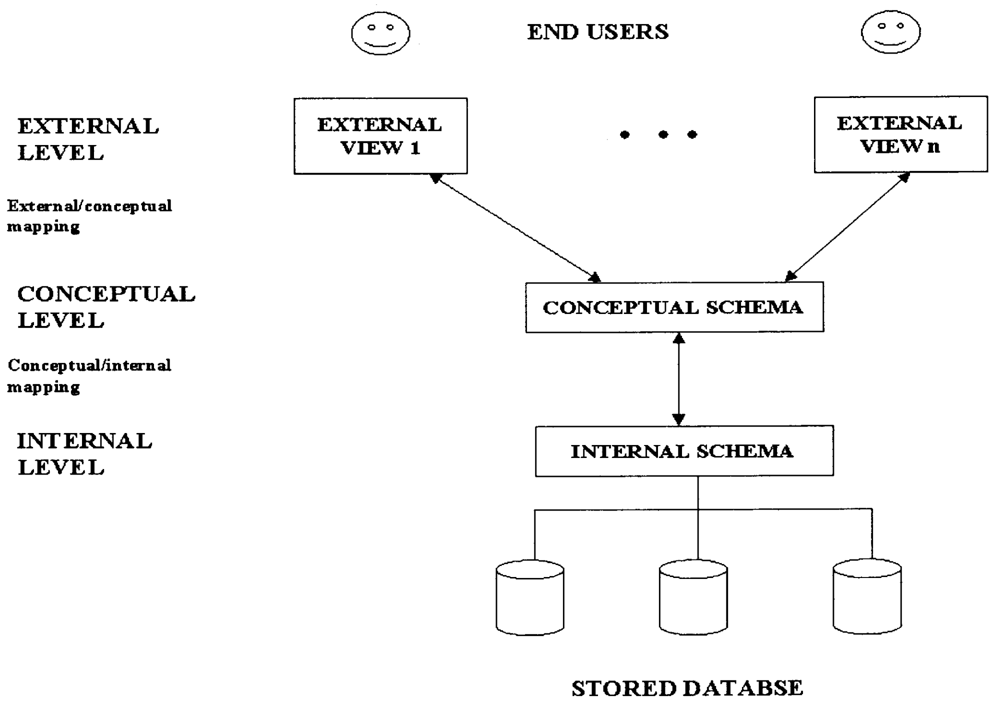


Fig. 4.1 The Three Schema Architecture [Elmasri, 94]

• The conceptual level has a conceptual schema, which describes the structure of the entire database for a community of users. The conceptual schema hides the details of physical storage structures and concentrates on describing entities, data type, relationships, user operations and constraints. A high-level data model or an implementation data model can be used at this level.

• The external or view level includes a number of external schemas or user views. Each external schema describes the part of a database that a particular user group is interested in and hides the rest of the database from that user group. A high-level data model or an implementation data model can be used at this level.

4.2 Introduction to Relational Database

The Normalized Relational Database model is adopted to design MLCA database. A *Normalized Relational Database* is a particular method of database design with a rigid set of rules. Strict adherence to database design rules is essential for providing the information their requires in just about any form and level of detail necessary; while allowing for ease of change as future requirements become known. Following the rules guarantees that there will be a minimum of data redundancy and the costs associated with storing and maintaining redundant data.

The design rules for a Normalized Relational Database can be summarized as follows:

- The tables in a database may only contain information specifically about the particular entity.
- Each table must have a unique *primary key* so that one occurrence of the entity can be distinguished from another. The primary key must be one of the columns in the row and may be composed of several columns.
- The order of the rows and the order of the columns within a row are unimportant because they can be viewed in any order desired.

- The intersection of a row and a column contains a piece of information, this intersection point is sometimes called an *attribute* An attribute cannot be subdivided [thus the term *atom*]; it is always entered and presented as a whole, but it may be formatted according to preference.
- All information in a particular column must be of the same nature, varying only in the way expected because of the particular row.
- No two columns in the same row may contain the same type of information; said another way: repeating fields are not allowed.
- Information in a particular column must be solely and completely dependent on the primary key of the given row. An exception is permitted in the case of a *foreign key* which contains the primary key of a row in another related table. In fact, "foreign keys" are the way tables are related to each other.

4.3 MLCA Database Requirements

4.3.1 MLCA Users

There are three classes of users of MLCA integrated database: Product/Process Designers, Demanufacturers, and MLCA Analysts. All users have some common tasks and some are specialized to their roles.

- Product/Process Designers

Designers must look forward in time to maximize the product's end-of-life yield of assemblies, parts and materials while looking backward to the world of existing products for feedstock sources for the current design. Lifecycle assessment (LCA) needs to be used as a mechanism for quantifying the impact of design and production issues over the working life of a product. To satisfy these requirements, the database should contain the following information:

a) Tree structure of all the components in a product.

b) Complete description of part, subassembly and final subassembly.

c) Part's material, weight, quantity, manufacturing process and geometry information.

d) Feedback from demanufacturers about the demanufacturability of the product.

- Demanufacturers

In the demanufacturing stage, the decision about reengineering, remanufacturing and reuse is made. Demanufacturers are interested in taking care of the process plan of demanufacturing, and optimization of that plan to get the maximum benefit. To satisfy these requirements, one needs information flow from the design to demanufacturing stage in the database. Hence, the database should contain the following information:

a) Information about demanufacturing facilities.

b) The connectivity between two parts and how they are connected.

c) The time required to disassemble the parts.

d) The tools needed to perform the disassembly.

- MLCA Analysts

Analysts are interested in all the stages of a product. They postulate and analyze the "what-if" scenarios. They determine the environmental performance of a product and identify changes in that product or process to improve its environmental footprint such that it is economically better and of higher quality. To satisfy these requirements and perform analysis, the database should contain the following information:

a) Material extraction and synthesis stage details.

b) Manufacturing process details.

c) Packaging and transportation details.

d) Use stage details.

e) Remanufacturing details.

g) Reengineering stage details.

4.3.2 Entities for ER model of MLCA database

An entity is defined as "the thing that can be distinctly identified" [Chen, 1976]. A group of similar entities form an entity set. Entities have properties, called attributes, which associate a value from the domain of values for that attribute with each entity in an entity set. The selection of relevant attributes for the entity set is a crucial step in the design of a real world model. An attribute or a set of attributes whose values uniquely identify each entity in an entity set is called a key for that entity. For the MLCA integrated database, the main entities are component, material, process, and facility. MLCA key factor is a weak entity as its existence fully depends upon on the process entity.

■ **COMPONENT**

A product is composed by different types of component. The basic types of component are parts and subassemblies. Thus with entity type COMPONENT itself as the superclass for all components, entities type PART and ASSEMBLY are its direct subclasses. However, there are three different types of assembly: subassembly, finalsubassembly and product. Thus entity type ASSEMBLY itself is a superclass which includes PRODUCT, SUBASSEMBLY and FINALSUBASSEMBLY. All those entities are meaningful and need to be represented explicitly because of their significance to the database applications. Hence COMPONENT, PART, ASSEMBLY, PRODUCT, SUBASSEMBLY and FINALSUBASSEMBLY form a three level hierarchy strucuture. The subclass inherits all the super-class properties. This subclass-superclass relationship is illustrated in Fig. 4.2.

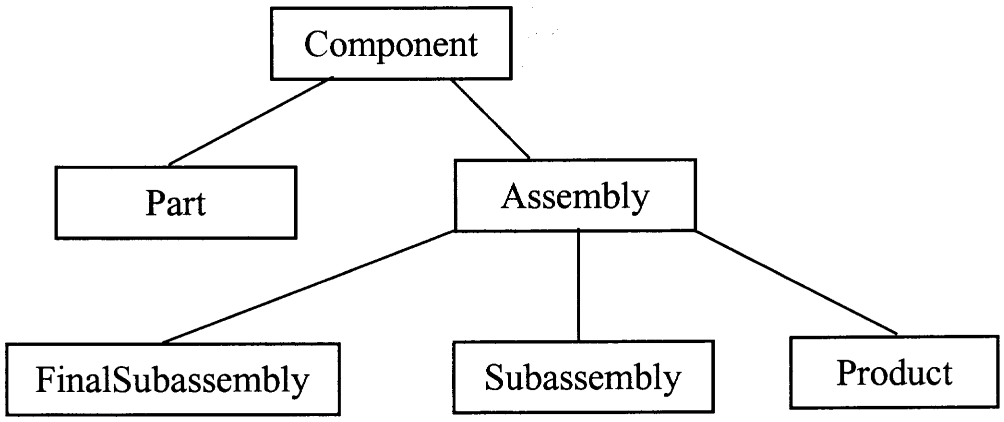


Fig. 4.2 Superclass COMPONENT with its subclass

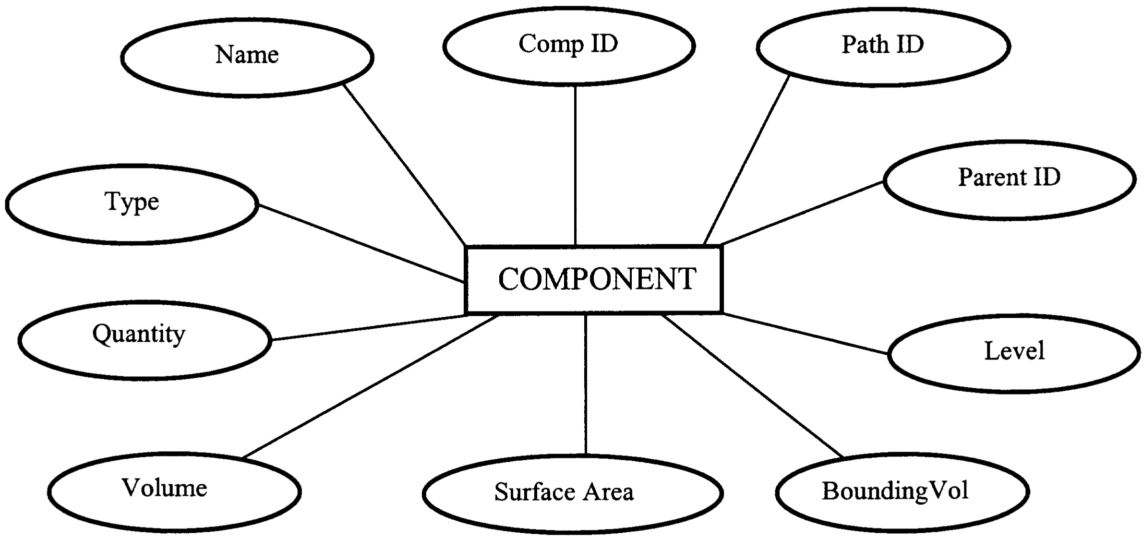


Fig. 4.3 Entity COMPONENT with its attributes

For an entity type COMPONENT, the attributes are a unique component identity number via *Comp_ID, name, Type, Quantity, Path ID, Parent ID, Level, Surface Area, Volume and Bounding Volume*. The entity component with its attributes is shown in Fig. 4.3. The attribute *Type* differentiates between the PART, SUBASSEMBLY, FINAL-SUBASSEMBLY and PRODUCT.

■ MATERIAL

In MLCA, material is the key issue needed to be considered through all stages of life of a product. The environmental effect of a product with the choice of different material also should be fedback to the designer for comparison. The attributes of the entity type MATERIAL are unique material identity number *Material_ID, Description, cost, Weight, Composition, Density and Environment Burden*. The entity MATERIAL with its specialized attributes is shown in Fig. 4.4.

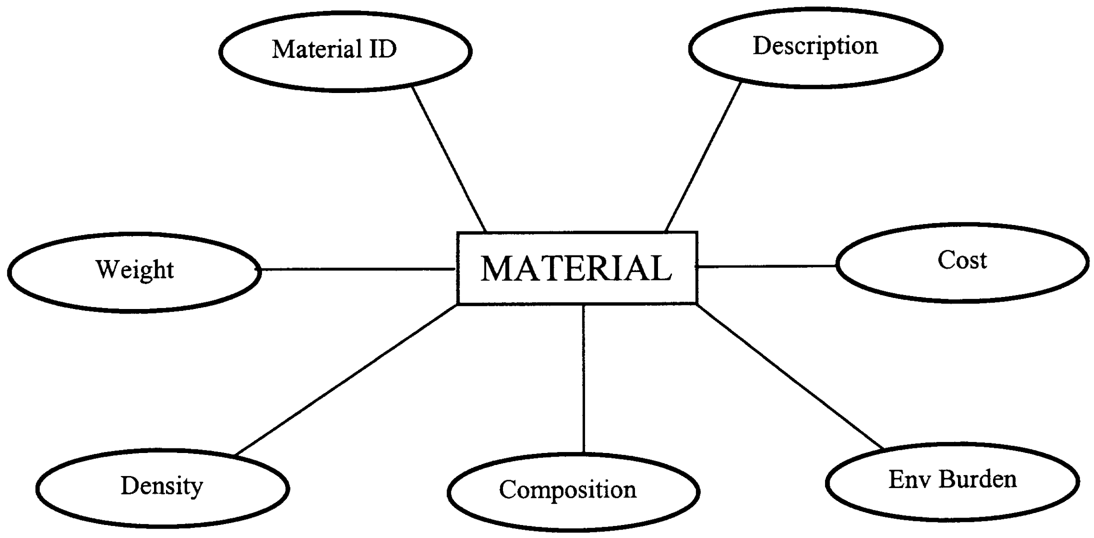


Fig. 4.4 Entity MATERIAL with its attributes

■ FACILITY

Facilities are used to process or transport different types of components. It puts significant effect on variable environmental issues, such as energy consumption and waste emission. Facilities are also closely related to the different choices of material. In MLCA database, the facilities are demanufacturing, manufacturing and transportation facilities. To represent any kind of facility, a super-class named FACILITY is formed. This FACILITY class has attributes that are common to all facility and thus this class is formed by the generalization method. The subclass and super-class relationship is shown in Fig. 4.5. The subclasses for Facility class are DEMANUFACTURING FACILITY, DISTRIBUTION FACILITY and MANUFACTURING FACILITY.

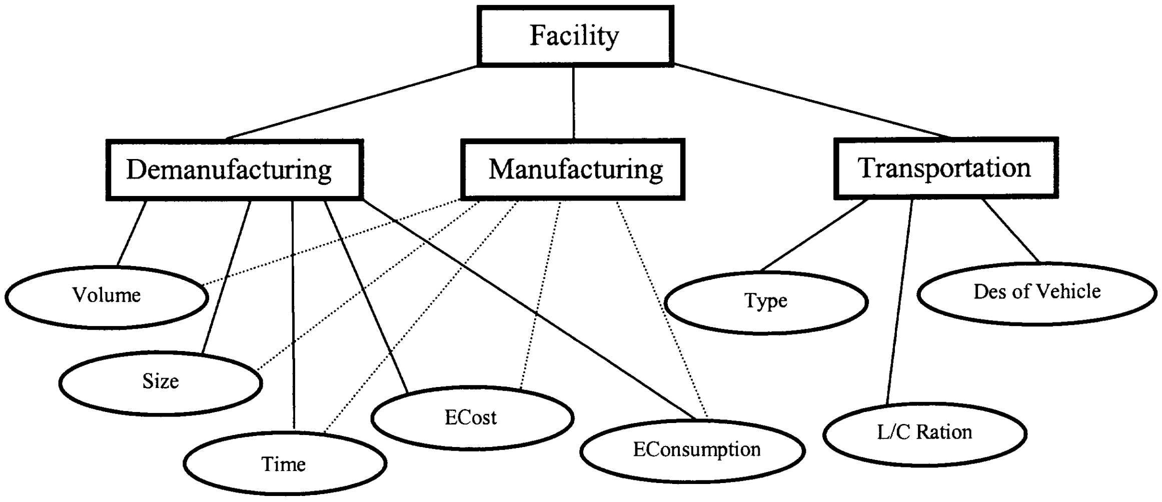


Fig. 4.5 Entity FACILITY with its subclasses

■ PROCESS

Different from traditional LCA software, MLCA holds process information using more stages: Use, material Processing, Demanufacturing, Remanufacturing, and Reengineering. The subclass and superclass relationship for the entity PROCESS is shown in Fig. 4.6.

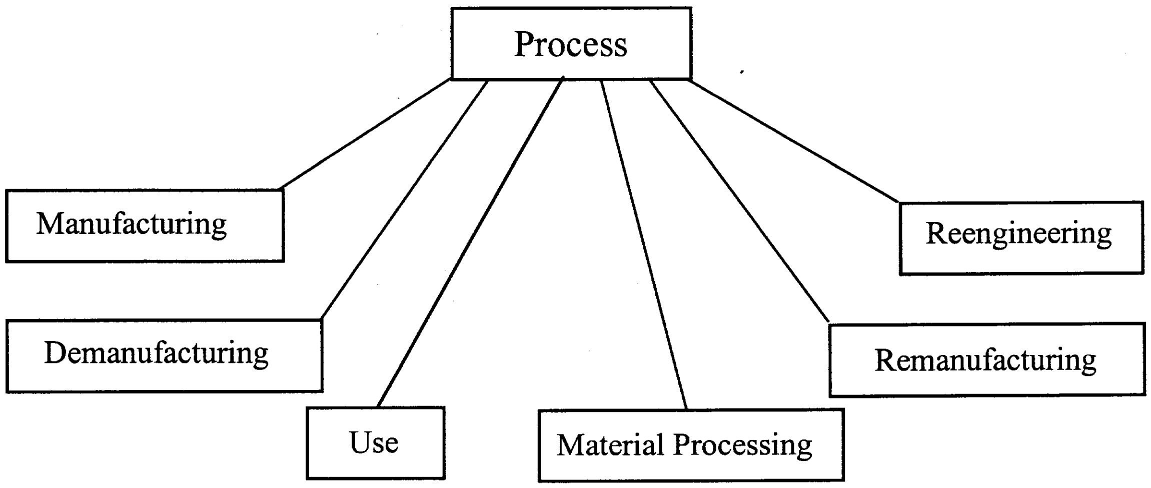


Fig. 4.6 Entity PROCESS with its subclasses

■ **MLCA Key Factors**

The outcome of each process can be referred by environmental burdens produced and energy and cost associated with that process. To represent this outcome, an entity named MLCA KEY FACTOR is formed. This is a weak entity as its existence depends upon a process. The super-class MLCA KEY FACTOR is formed with its subclass POLLUTION INGREDIENTS, ENERGY and COST, which is shown as Fig. 4.7.

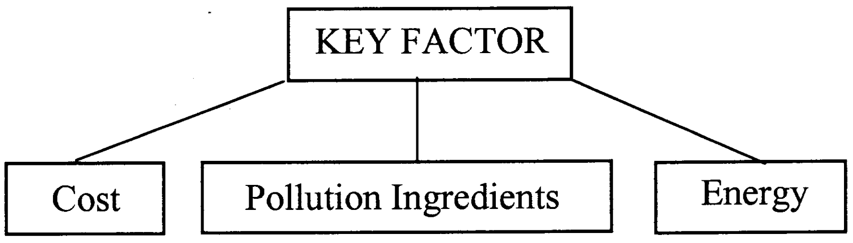


Fig. 4.7 Superclass MLCA Key Factor with its subclass

4.3.3 Relationships of MLCA Database

A relationship type R among n entity types $E_1, E_2, \ldots,$ and E_n defines a set of associations among entities from this type. Mathematically, R is a set of relationship instances r_i, where each r_i associates n entities $(r_i, e_{i1}, e_{i2}, \ldots, e_{in})$, and each e_{ij} in r_i is a member of entity type E_j, $1 \leq j \leq n$. Informally, each relationship instance r_i in R is an association of entities, where the association includes exactly one entity from each participating entity type.

Each entity type that participates in a relationship type plays a particular role in the relationship. The relationship name signifies the role that a participating entity plays in the relationship instance. Relationship types usually have certain constraints that limit the possible combinations of entities that may participate in relationship instances. These constraints are determined from the miniworld situation that the relationships represent. The cardinality ratio specifies the number of relationship instances in which an entity can participate. For the MLCA database, the following relationships are specified.

- **Made by**

This relationship represents the connection between the entities PART and MATERIAL and also between FINALSUBASSEMBLY and MATERIAL. Every PART and FINALSUBASSEMBLY can be composed of at least one MATERIAL and up to a maximum of 'n'. Similarly a MATERIAL can be used in zero component or up to a maximum of 'n' components. Hence cardinality ratio is 1:n. This relationship has attribute *weight of material* used for that PART.

- **Processed by**

This relationship represents the connection between PART/FINALSUBASSEMBLY and PROCESS and also between the MATERIAL AND PROCESS. Every

PART/FINALSUBASSEMBLY undergoes at least one PROCESS and at most ‘n’. Each PROCESS can be used by minimum zero PART/FINALSUBASSEMBLY to maximum ‘n’ PART/FINALSUBASSEMBLY. Similarly each MATERIAL can undergo maximum ‘n’ processes and each PROCESS can use upto a maximum of ‘n’ MATERIAL.

■ **Used by**

Every facility can be used by a specific process and upto a maximum of ‘n’ processes. This *Used by* relation represents the connection between PRODUCTION PROCESS and PRODUCTION FACILITY, PACKAGING PROCESS and PACKAGING FACILITY, DEMANUFACTURING PROCESS and DEMANUFACTURING FACILITY, REMANUFACTURING PROCESS and REMANUFACTURING FACILITY and also BETWEEN REENGINEERING PROCESS and REENGINEERING FACILITY. This relationship has attribute time that the facility is used for that process.

■ **Produces**

This relationship represents the connection between PROCESS and MLCA KEY FACTOR. It is 1:1 relationship, as each PROCESS produces MLCA KEY FACTORS. Also each MLCA KEY FACTORS is associated with one PROCESS [Suratran, 2000].

4.4 ER Schema for MLCA

After Entity and Relationship are defined, we can create the Entity-Relationship diagram (ER diagram) for the MLCA database. The emphasis of ER diagram is on representing schema rather than the instances. The ER schema for the MLCA database is shown in Fig. 4.8.

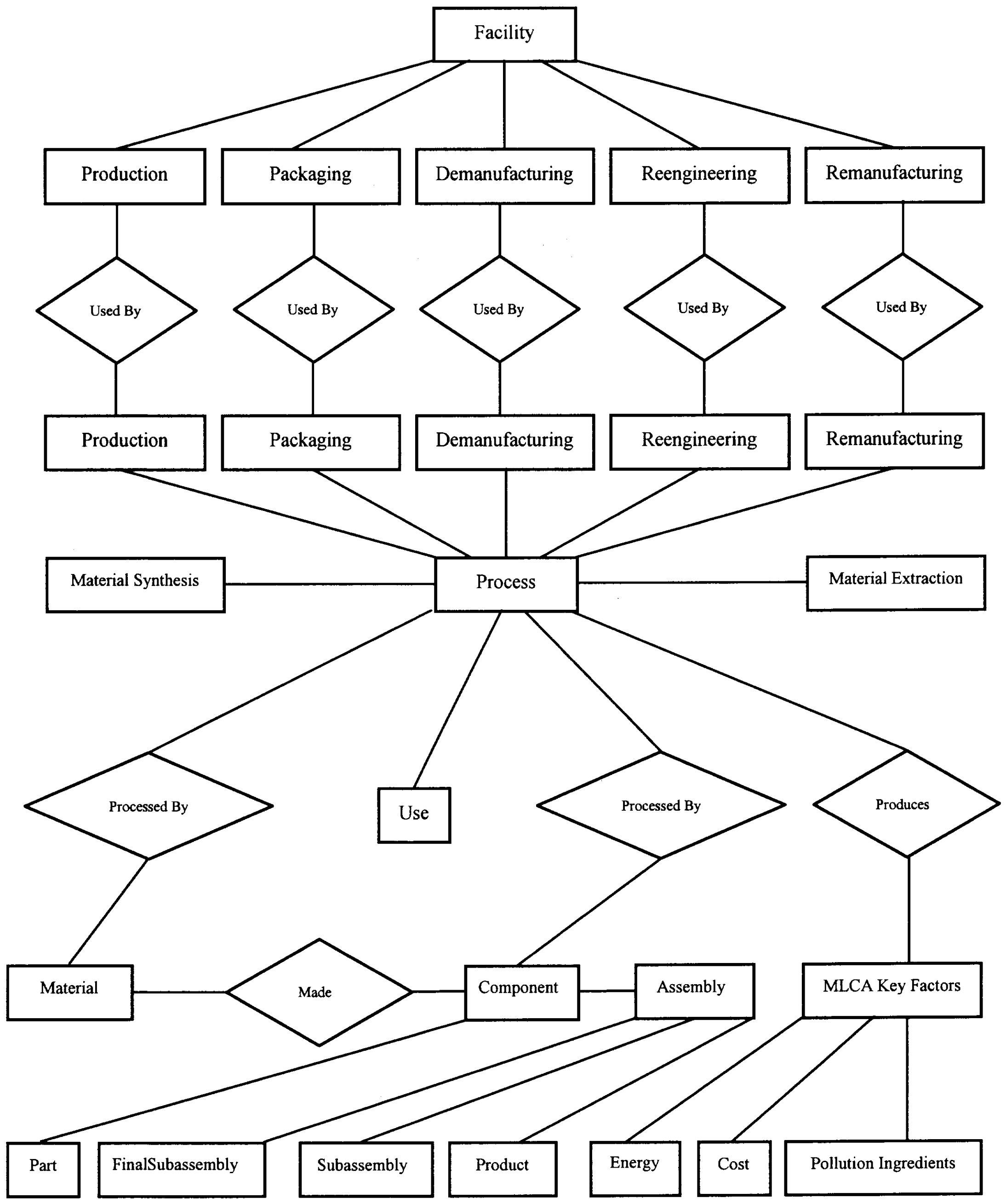


Fig. 4.8 ER diagram for the MLCA database

4.5 MLCA Database Design using ER-to-Relational Mapping

Based on the ER model in the previous section, the design of a relational database for MLCA software can be finished by using ER-to-relational mapping. Thus the MLCA relational database can be created, which partly list as follows.

PROJECTS

Project_ID	Project_Description

PROJECT_ASSEMBLIES

Project_ID	**Path_ID**	Name	Description	Parent_ID	Quantity
Type	Level				

SUB_MATERIAL

Project_ID	**Path_ID**	**Material_ID**	Weight	WeightUnit	Volume
VolumeUnit	SurfaceArea	AreaUnit	BVUnit	BoundingVol	

MATERIALS

Material_ID	Description	Density	DensityUnit

PROJECT_MODES

Project_ID	**Mode_ID**	PwoerConsumption	PowerUnit_ID	Energy_Price
Energy_ID	ModeUti_coeff	UtilizationFactor		

CHAPTER 5

IMPLEMENTATION DETAIL WITH CASE STUDIES

To design an integrated DFE software package, a suitable CAD software tool is needed to combine with MLCA software. There are many popular CAD software packages available, e.g., CATIA, I-DEAS and Pro/ENGINEER. After the comparison of different CAD packages, Pro/ENGINEER is chosen as a CAD package for developing CAD-MLCA interface, as it is the most commonly used CAD software in electronic products and has a very good application programmer interface (API). Pro/TOOLKIT is the customization toolkit for Pro/ENGINEER from Parametric Technology Corporation. It gives customers and third-party developers the ability to add functionality to Pro/ENGINEER by writing code in the C programming language and integrating the resulting application into Pro/ENGINEER in a seamless way. Pro/TOOLKIT also provides a large library of C functions that enable the external application to access the database and applications of Pro/ENGINEER in a controlled and safe manner.

5.1 Introduction to Pro/ENGINEER

5.1.1 The basic characteristic of Pro/ENGINEER

Pro/ENGINEER provides mechanical engineers with an approach to mechanical design automation based on the solid modeling technology and the following features.

■ 3-D Modeling

The essential difference between Pro/ENGINEER and traditional CAD systems is that the models created in Pro/ENGINEER exist as three dimensional solids. Other 3-D modelers represent only the surface boundaries of the model. Pro/ENGINEER models the complete solid. This not only facilitates the creation of realistic geometry, but also allows for accurate model calculations, such as those for mass properties.

■ **Parametric Design**

Dimensions such as angle, distance, and diameter control Pro/ENGINEER model geometry. Users can create relationships that allow parameters to be automatically calculated based on the value of other parameters. When users modify the dimensions, the entire model geometry can update according to the relations they created.

■ **Feature-Based Modeling**

Users create models in Pro/ENGINEER by building features. These features have intelligence, in that they contain knowledge of their environment and adapt predictably to change. Each feature asks a user for specific information based on the feature type. For example, a hole has a diameter, depth, and placement, while a round has a radius and edges to round.

■ **Associativity**

Pro/ENGINEER is a fully-associative system. This means that a change in the design model anytime in the development process is propagated throughout the design, automatically updating all engineering deliverables, including assemblies, drawings, and manufacturing data. Associativity makes concurrent engineering possible by encouraging change, without penalty, at any point in the development cycle. This enables downstream functions to contribute their knowledge and expertise early in the development cycle.

5.1.2 Parent-Child Relationships

The definition of a feature frequently relies on dimensional and geometric cues taken from another feature. This kind of relationship is termed a *parent-child relationship*. The parent-child relationship is one of the most powerful aspects of Pro/ENGINEER. When a parent feature is modified, its children are automatically recreated to reflect the changes in the geometry of the parent feature. It is therefore essential to reference feature dimensions and geometry so design modifications are correctly propagated throughout the model. Because of children reference parents, features can exist without children, but children cannot exist without their parents.

5.2 Introduction to Pro/TOOLKIT

Pro/TOOLKIT is the customization toolkit for Pro/ENGINEER from Parametric Technology Corporation (PTC). Pro/TOOLKIT is the PTC application programmer's interface (API). Pro/TOOLKIT replaces, and also contains, Pro/DEVELOP, the customization toolkit in earlier versions of Pro/ENGINEER. Pro/TOOLKIT uses a new and more consistent "object-oriented" style than Pro/DEVELOP. Many of the library functions familiar to experienced users of Pro/DEVELOP have been replaced by equivalent Pro/TOOLKIT functions.

5.2.1 Pro/TOOLKIT Style

Pro/TOOLKIT uses a strongly object-oriented style in which the data structures that transfer information between Pro/ENGINEER and the application are not directly visible to the application, but are accessible only through the use of Pro/TOOLKIT functions.

Pro/TOOLKIT therefore tends to have more functions, but fewer explicit data structures, than Pro/DEVELOP does.

5.2.2 Objects and Actions

The most basic concepts in Pro/TOOLKIT are objects and actions. Each C function in the Pro/TOOLKIT library performs an action on an object of a specific type. The Pro/TOOLKIT convention for the name of each function is the prefix "Pro", the name of the object type, and then the name of the action it performs.

A Pro/TOOLKIT object is a well-defined and self-contained C structure that you can use to perform actions relevant to that object. Most objects are items in the Pro/ENGINEER database, such as features and surfaces. Others, however, are more abstract or transient, such as the information resulting from a *select* action.

In Pro/TOOLKIT, each type of an object has a standard name consisting of a capitalized word that describes the object. Pro/TOOLKIT provides a C *typedef* for each object used for variables and arguments that refer to those objects. Pro/TOOLKIT objects have a hierarchical relationship that reflects the hierarchy of the Pro/ENGINEER database. For example, a *Feature* object can contain, among others, objects of type *Surface*. Also, some Pro/TOOLKIT functions require names that include more than one object type. Another convention for Pro/TOOLKIT functions is that the first argument always identifies the object, and the input arguments come before the output arguments.

5.2.3 Function Prototyping

Each Pro/TOOLKIT function is provided with an ANSI function prototype. The C compilers on all the platforms supported by Pro/TOOLKIT provide at least the option of function prototype checking. All the functions for a particular Pro/TOOLKIT object are prototyped in a header file whose name is the name of that object.

5.2.4 Function Error Statuses

The return type of most Pro/TOOLKIT functions is *ProError*. *ProError* is an enumerated type that has a value for each of the most common cases where Pro/TOOLKIT functions can succeed or fail. The normal value for success is PRO_TK_NO_ERROR. The other "failure" statuses can occur when there is a genuine problem, or for more benign reasons. Therefore, users should take care about how their program reacts to the status returned from a Pro/TOOLKIT function - there might be several types of failure and success that need to be handled in different ways.

5.2.5 Working Principle of Pro/TOOLKIT

The standard method by which Pro/TOOLKIT application code is integrated into Pro/ENGINEER is through the use of dynamically linked libraries (DLLs). When users compile their Pro/TOOLKIT application C code and link it with the Pro/TOOLKIT library, they create an object library file designed to be linked into the Pro/ENGINEER executable when Pro/ENGINEER starts up. This method is referred to as "DLL mode".

Pro/TOOLKIT also supports a second method of integration: the "multiprocess", or spawned mode. In this mode, the Pro/TOOLKIT application code is compiled and linked to form a separate executable. This executable is designed to be spawned by Pro/ENGINEER and runs as a child process of the Pro/ENGINEER session. In the DLL mode, the exchanges between the Pro/TOOLKIT application and Pro/ENGINEER are made through direct function calls. In the multiprocess mode, the same effect is created by an interprocess messaging system that simulates direct function calls by passing the information necessary to identify the function and its argument values between the two processes.

Users can use a Pro/TOOLKIT application in either DLL mode or multiprocess mode without changing any of the C source code in the application. The methods of setting the mode are described in detail later in this chapter. It is also possible to use more than one Pro/TOOLKIT application within a single session of Pro/ENGINEER, and these can use any combination of modes.

5.3 Database Connection of Pro/ENGINEER and MLCA

As mentioned in Chapter 2, CAD software and MLCA software share a common portion of data which includes the design information of a product. The design information plays an essential role in the integrated DFE software package and need to be populated to MLCA database. If a product is very huge or complicated, say a tanker, there are millions of components associated with this product and every single component itself also has many data items. Thus the total number of data items needed to populate is incredibly huge. Obviously, it is not only time-consuming but also error prone to input this kind of information to MLCA database manually. To avoid this kind of depressed jobs, we can connect Pro/ENGINEER database with MLCA database by using library functions provided by Pro/Toolkit. Thus a utility part for the integrated DFE software is designed which can extract design information of a product from Pro/Engineer and populate to MLCA database automatically.

5.4 Interface Design

There are two steps in the development of the CAD-MLCA interface. The first is to develop a Pro/TOOLKIT application and the next is to implement an MLCA database population program. The relationships among Pro/Engineer, the interface and MLCA software are illustrated in Fig. 5.1.

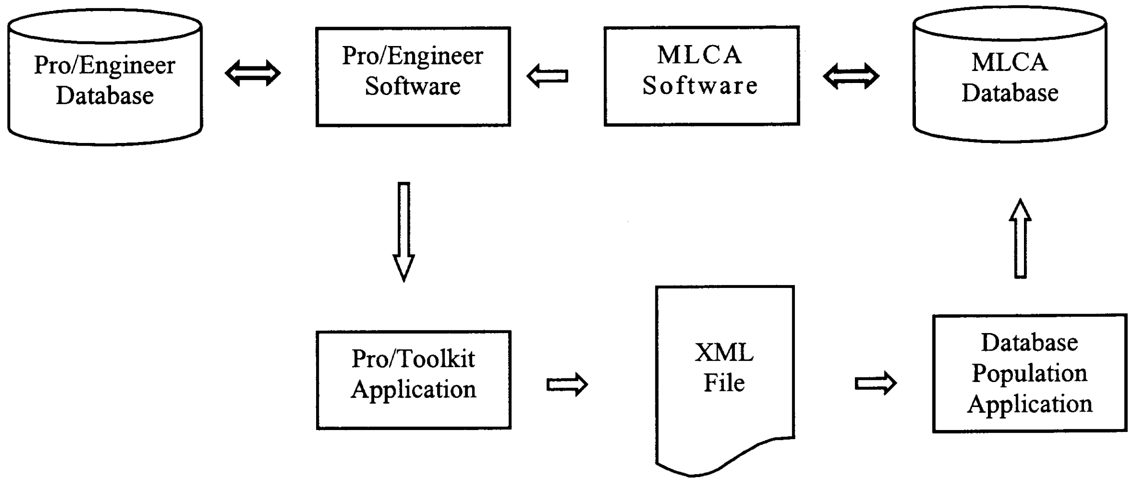


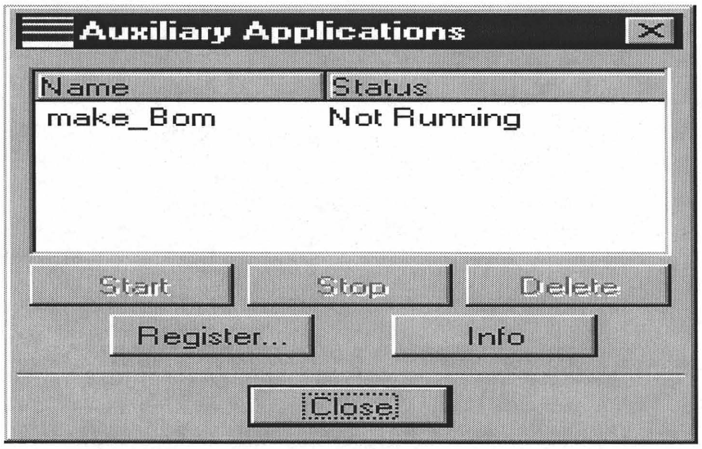
Fig. 5.1 Design of CAD-MLCA interface

The Pro/TOOLKIT application program extracts the product information, which is modeled in Pro/ENGINEER software, from the Pro/ENGINEER database. This application creates a flat file that contains the data information needed for MLCA in a defined XML file format. The second part contains an application 'Database Population Program', which connects to and populates the MLCA database. MLCA software uses the MLCA database to perform multi-lifecycle assessment of the product and feedback the result.

5.4.1 Register the Application Program

The developed interface is Internet web-based. After downloading the Pro/TOOLKIT application for MLCA from http://www.merc.njit.edu/interface, the user can start Pro/ENGINEER software. To run the application program, the user needs to provide information to Pro/ENGINEER about the files that specify the Pro/TOOLKIT application. To do this, create a small text file 'protk.dat', called the Pro/TOOLKIT 'registry file', that Pro/ENGINEER finds and reads. Pro/ENGINEER allows different version of registry files to exist on the system at the same time. That means a user can create different applications and put them to different directories of the same computer. Accordingly, there are two different ways to a register user's application related to two different cases.

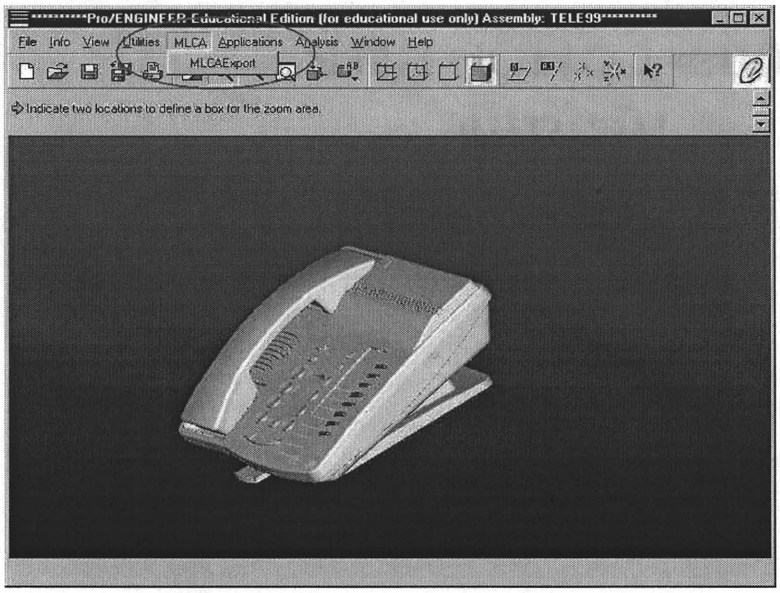
In the first case, users can put the registry file to directory which is the same as Pro/TOOLKIT application file. While Pro/ENGINEER software is activated, it finds the protk.dat file in the same directory and takes it as a default registry file.



Name	Status
make_Bom	Not Running

Fig. 5.2 Register application for Pro/Engineer

In another case, users can put the registry file anywhere they want. After Pro/ENGINEER is running and the main window show up on the screen, they can click on the menu #utilities and then #Auxiliary applications under #utilities from the menu bar. The screen shown in Fig. 5.2 is displayed.


**********Pro/ENGINEER Educational Edition (for educational use only) Assembly: TELE99**********
File
Info
View
Utilities
MLCA
Applications
Analysis
Window
Help
MLCAExport

Indicate two locations to define a box for the zoom area.

Fig. 5.3 Customized window for MLCA application

Users are required to register their application first by clicking on button 'Register' and then starting the application. Once the application is started, they can find a new menu item *'MLCA'* that has been added to Pro/ENGINEER default menu. The new menu item is between the menu *'Utilities'* and *'Applications'* which is shown in Fig. 5.3.

5.4.2 Extract Database from Pro/ENGINEER

With this customized menu, the user just need to click the *'MLCAExport'* button, the Pro/TOOLKIT application program automatically extracts the design information about the loaded product, say *TEL99*, from Pro/ENGINEER database and simultaneously convert it to two identical files *tel99.xml* and *MLCA.xml* with XML file format. The function invoked by the menu has implemented a recursive algorithm for traversing the assembly. In the recursive algorithm, the checking condition is to check against component type. If the type is *part*, check for its geometry, material and mass properties. If the component is an *assembly*, traverse again all its sub-components.

While grabbing the information from the product's assembly file opened in the Pro/ENGINEER, the interface prompts users for the missing information. Thus, if users have not entered any material for a part in that assembly, the application prompts them to enter material information. As a result of Pro/TOOLKIT application, the newly created formatted XML files contain the information of the assembly and all of its sub-components, their name, process depth (level), and their parent component/assembly name. If the component is a part, its material, weight, density, surface area, volume, bounding volume and corresponding units are also written to the file.

5.4.3 Define a Formatted XML File

As shown in Fig. 5.1, a formatted XML file plays the critical role as the intermediate file to connect the databases of Pro/ENGINEER and MLCA. XML stands for Extensible Markup Language. XML is a meta-markup language that provides a format for describing structured data. This facilitates more precise declarations of content and more meaningful search results across multiple platforms. In addition, XML enables a new generation of Web-based data viewing and manipulation applications.

XML provides a structural representation of data that can be implemented broadly and is easy to deploy. Once the data is on the client desktop, it can be manipulated, edited, and presented in multiple views, without return trips to the server. Servers can now become more scalable, due to lower computational and bandwidth loads. Also, since data is exchanged in the XML format, it can be easily merged from different sources.

XML is a text-based format, similar to HTML in many respects, designed specifically to store and transmit data. An XML source is made up of XML elements, each of which consists of a start tag (<title>), an end tag (</title>), and the information between the two tags (referred to as the content). Like HTML, an XML document holds text annotated by tags. However, unlike HTML, XML allows an unlimited set of tags, each indicating not how something should look, but what something means. For example, an XML element might be tagged as a price, an order number, or a name. It is up to each document's author to determine what kind of data to use and which tag names fit best.

XML brings so much power and flexibility to Web-based applications that provides a number of compelling benefits to developers and users:

- More meaningful searches ;
- Development of flexible Web applications ;
- Data integration from disparate sources;
- Local computation and manipulation of data;
- Multiple views of the data; and
- Granular updates.

The core task in the implementation of the CAD-MLCA interface is to define the format of *MLCA.XML* file, which is created as a result of Pro/Toolkit application. This *MLCA.XML* file should be programmatically readable. Each record holds the information of a single component with a definite format as follows.

```
<Component>
<Level level="1" />
<Type type="PRT" />
<Name name="ROUND-BUTTON" />
<Parent parent="TELE99" />
<Material material="RUBBER" />
<LengthUnit lengthunit="MM" />
<MassUnit massunit="KILOGRAM" />
<Weight weight="0.000541" />
<Density density="1.260000e-006" />
<SurfaceArea area="326.184926" />
<Volume volume="429.463570" />
<BoundingVolume volume="622.080975" />
</Component>
```

To add any new attribute to CAD-MLCA interface, one new entry will be added to the record format without disturbing the population program and thus this format is extensible.

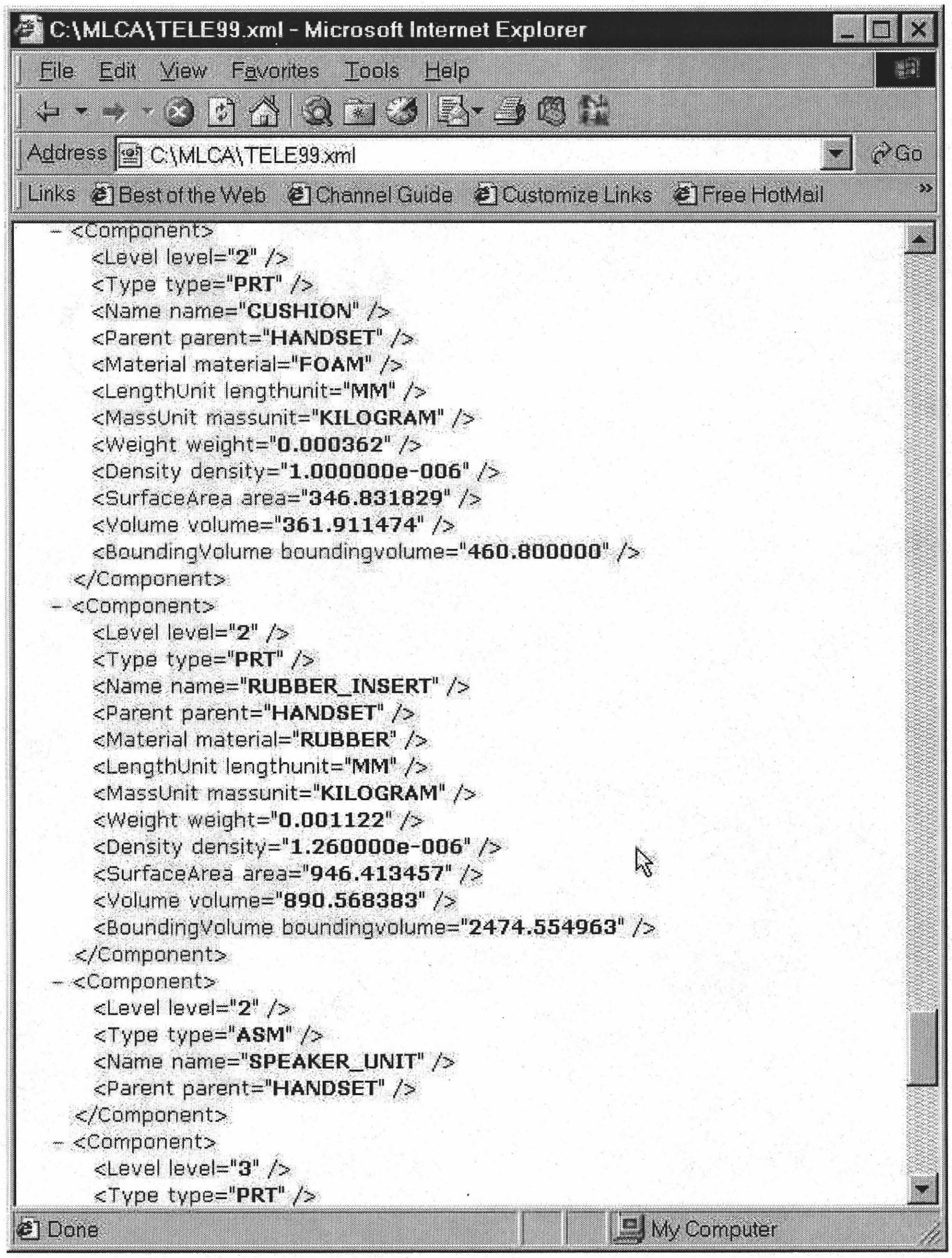
5.5 Automatic Population for MLCA Database

After being uploaded on the MERC server, the formatted XML file is checked and the database population program populates the database with the data in the XML file. Thus, as the output of this interface, a user has its product description data populated without any manual entry in MLCA software and is ready to perform environmental analysis of the product on the MLCA software.

5.6 Case Study: TEL99

As a case study of the CAD-MLCA interface, a typical office telephone of Lucent Technologies is chosen as the first example. It is designed and manufactured in 1999 (TEL99) and shown in Fig. 5.4. TEL99 is a so-called 'perfect' product sample for the interface, which means that its design information from CAD software completely satisfies the requirements of all MLCA stages.

Fig. 5.4 Telephone 1999 (TEL99) of Lucent Technologies

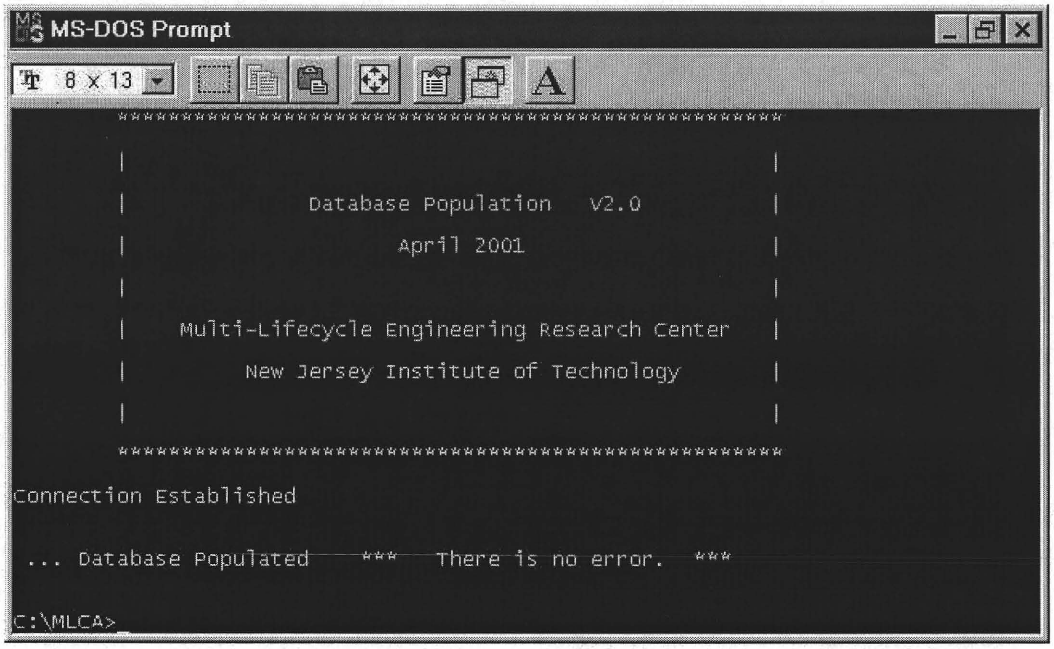

C:\MLCA\TELE99.xml - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address C:\MLCA\TELE99.xml
Go
Links Best of the Web Channel Guide Customize Links Free HotMail
Done
My Computer

```
- <Component>
    <Level level="2" />
    <Type type="PRT" />
    <Name name="CUSHION" />
    <Parent parent="HANDSET" />
    <Material material="FOAM" />
    <LengthUnit lengthunit="MM" />
    <MassUnit massunit="KILOGRAM" />
    <Weight weight="0.000362" />
    <Density density="1.000000e-006" />
    <SurfaceArea area="346.831829" />
    <Volume volume="361.911474" />
    <BoundingVolume boundingvolume="460.800000" />
  </Component>
- <Component>
    <Level level="2" />
    <Type type="PRT" />
    <Name name="RUBBER_INSERT" />
    <Parent parent="HANDSET" />
    <Material material="RUBBER" />
    <LengthUnit lengthunit="MM" />
    <MassUnit massunit="KILOGRAM" />
    <Weight weight="0.001122" />
    <Density density="1.260000e-006" />
    <SurfaceArea area="946.413457" />
    <Volume volume="890.568383" />
    <BoundingVolume boundingvolume="2474.554963" />
  </Component>
- <Component>
    <Level level="2" />
    <Type type="ASM" />
    <Name name="SPEAKER_UNIT" />
    <Parent parent="HANDSET" />
  </Component>
- <Component>
    <Level level="3" />
    <Type type="PRT" />
```

Fig. 5.5 TELE99.xml

After running the Pro/Toolkit application program, the design information related to all components of TEL99 is extracted and stored into TEL99.xml with XML format. At the same time, the process depth (level) for each component is also calculated by using a recursive algorithm. The extracted TEL99.xml is shown in Fig.5.5.

All the data items associated with a component are 'absolutely' accurate except for 'boundingvolume'. The value for the bounding volume of the same component can be variable depending on the coordinate system a designer uses.

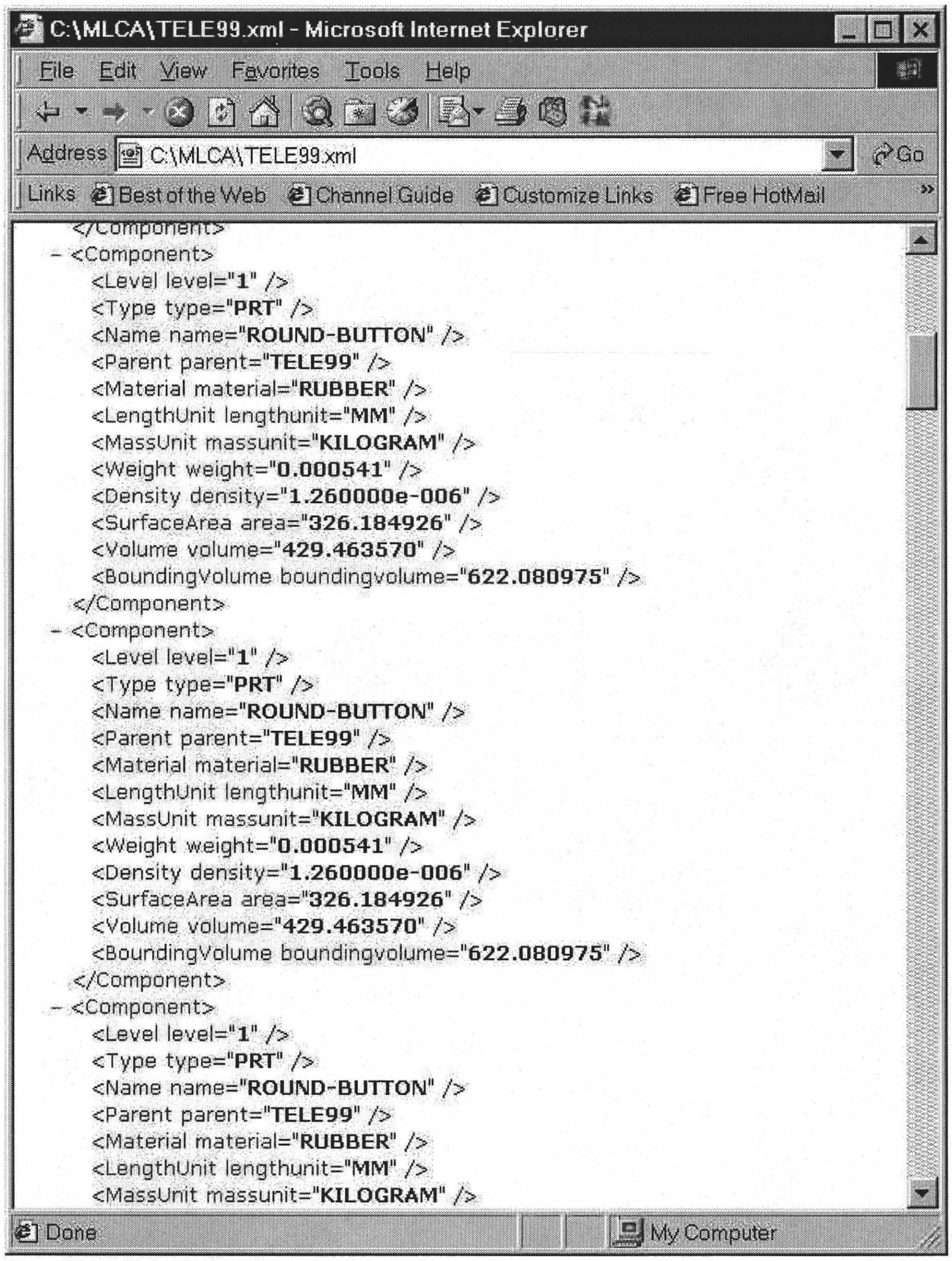


```
        ****************************************************
        |                                                  |
        |              Database Population   V2.0          |
        |                     April 2001                   |
        |                                                  |
        |    Multi-Lifecycle Engineering Research Center   |
        |         New Jersey Institute of Technology       |
        |                                                  |
        ****************************************************
Connection Established

 ... Database Populated    ***   There is no error.   ***

C:\MLCA>_
```

Fig. 5.6 The prompt window after the population of TELE99

After the TELE99.xml is stored, the population program can run. It traverses all the components for TELE99 and picks up all the data items necessary for the current version of MLCA software, which is developed under Windows 98 by using Visual Basic 6.0 and Microsoft Access 2000. The component information is populated to the corresponding tables in MLCA.

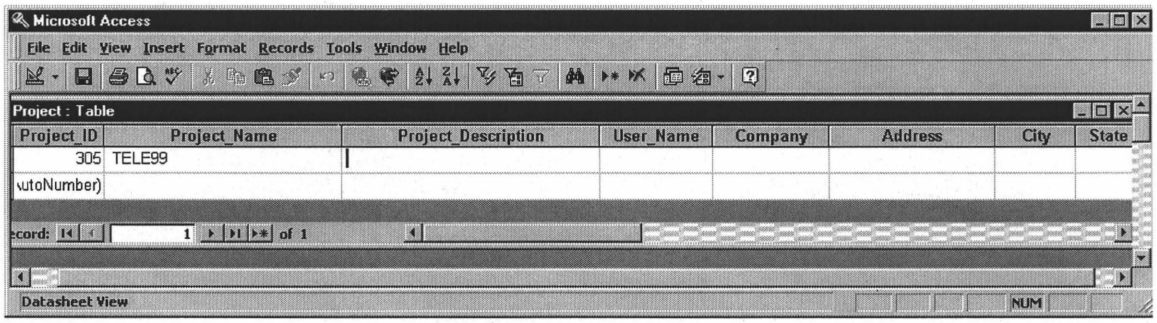
As mentioned earlier, TELE99 is a perfect sample for the current version of MLCA software without any loss of design information. Thus, as shown in Fig. 5.6, the population program prompts 'No Error' message after the population is finished.


C:\MLCA\TELE99.xml - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address C:\MLCA\TELE99.xml
Go
Links Best of the Web Channel Guide Customize Links Free HotMail
Done
My Computer

```
    </Component>
  - <Component>
      <Level level="1" />
      <Type type="PRT" />
      <Name name="ROUND-BUTTON" />
      <Parent parent="TELE99" />
      <Material material="RUBBER" />
      <LengthUnit lengthunit="MM" />
      <MassUnit massunit="KILOGRAM" />
      <Weight weight="0.000541" />
      <Density density="1.260000e-006" />
      <SurfaceArea area="326.184926" />
      <Volume volume="429.463570" />
      <BoundingVolume boundingvolume="622.080975" />
    </Component>
  - <Component>
      <Level level="1" />
      <Type type="PRT" />
      <Name name="ROUND-BUTTON" />
      <Parent parent="TELE99" />
      <Material material="RUBBER" />
      <LengthUnit lengthunit="MM" />
      <MassUnit massunit="KILOGRAM" />
      <Weight weight="0.000541" />
      <Density density="1.260000e-006" />
      <SurfaceArea area="326.184926" />
      <Volume volume="429.463570" />
      <BoundingVolume boundingvolume="622.080975" />
    </Component>
  - <Component>
      <Level level="1" />
      <Type type="PRT" />
      <Name name="ROUND-BUTTON" />
      <Parent parent="TELE99" />
      <Material material="RUBBER" />
      <LengthUnit lengthunit="MM" />
      <MassUnit massunit="KILOGRAM" />
```

Fig. 5.7. Components with identical information

As shown in Fig. 5.17, some components hold more than one identical sub-components. The population algorithm checks and counts all the components with identical information, implying that those are identical components, and populate once as a single component with their quantity.



Microsoft Access

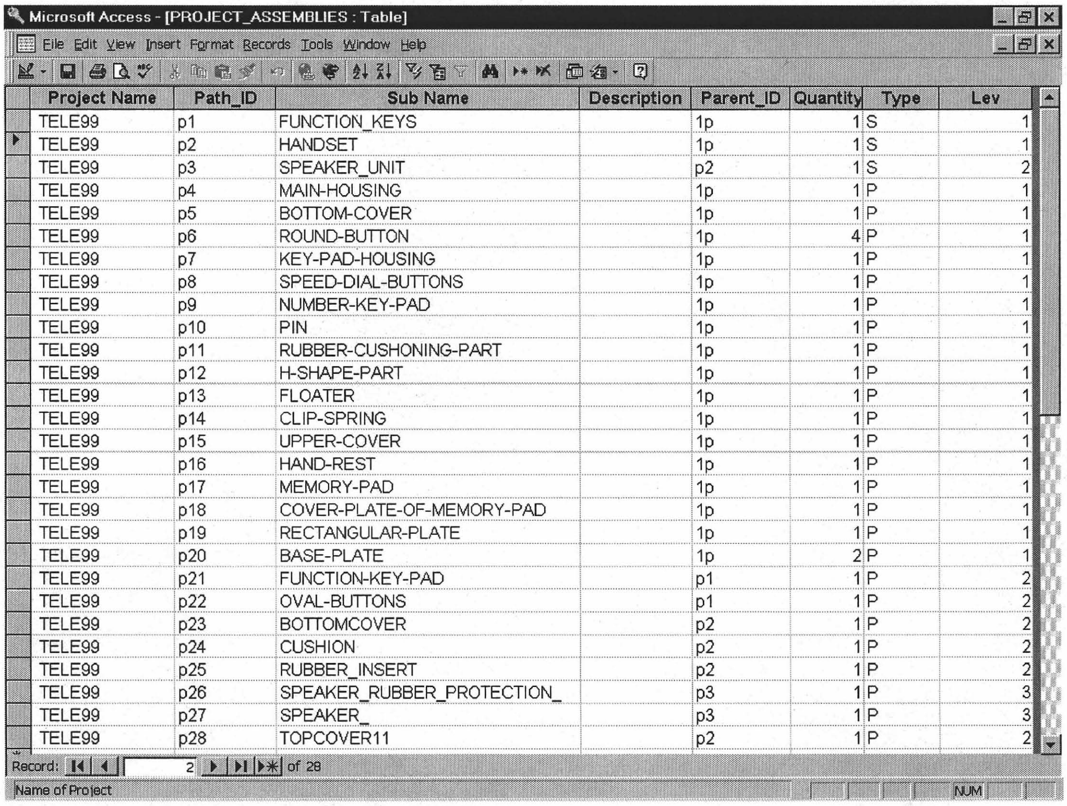
Project : Table

Project_ID	Project_Name	Project_Description	User_Name	Company	Address	City	State
305	TELE99						
utoNumber)							

cord: 1 of 1

Datasheet View

Fig. 5.8 Table PROJECT after population



Microsoft Access - [PROJECT_ASSEMBLIES : Table]

Project Name	Path_ID	Sub Name	Description	Parent_ID	Quantity	Type	Lev
TELE99	p1	FUNCTION_KEYS		1p	1	S	1
TELE99	p2	HANDSET		1p	1	S	1
TELE99	p3	SPEAKER_UNIT		p2	1	S	2
TELE99	p4	MAIN-HOUSING		1p	1	P	1
TELE99	p5	BOTTOM-COVER		1p	1	P	1
TELE99	p6	ROUND-BUTTON		1p	4	P	1
TELE99	p7	KEY-PAD-HOUSING		1p	1	P	1
TELE99	p8	SPEED-DIAL-BUTTONS		1p	1	P	1
TELE99	p9	NUMBER-KEY-PAD		1p	1	P	1
TELE99	p10	PIN		1p	1	P	1
TELE99	p11	RUBBER-CUSHONING-PART		1p	1	P	1
TELE99	p12	H-SHAPE-PART		1p	1	P	1
TELE99	p13	FLOATER		1p	1	P	1
TELE99	p14	CLIP-SPRING		1p	1	P	1
TELE99	p15	UPPER-COVER		1p	1	P	1
TELE99	p16	HAND-REST		1p	1	P	1
TELE99	p17	MEMORY-PAD		1p	1	P	1
TELE99	p18	COVER-PLATE-OF-MEMORY-PAD		1p	1	P	1
TELE99	p19	RECTANGULAR-PLATE		1p	1	P	1
TELE99	p20	BASE-PLATE		1p	2	P	1
TELE99	p21	FUNCTION-KEY-PAD		p1	1	P	2
TELE99	p22	OVAL-BUTTONS		p1	1	P	2
TELE99	p23	BOTTOMCOVER		p2	1	P	2
TELE99	p24	CUSHION		p2	1	P	2
TELE99	p25	RUBBER_INSERT		p2	1	P	2
TELE99	p26	SPEAKER_RUBBER_PROTECTION_		p3	1	P	3
TELE99	p27	SPEAKER_		p3	1	P	3
TELE99	p28	TOPCOVER11		p2	1	P	2

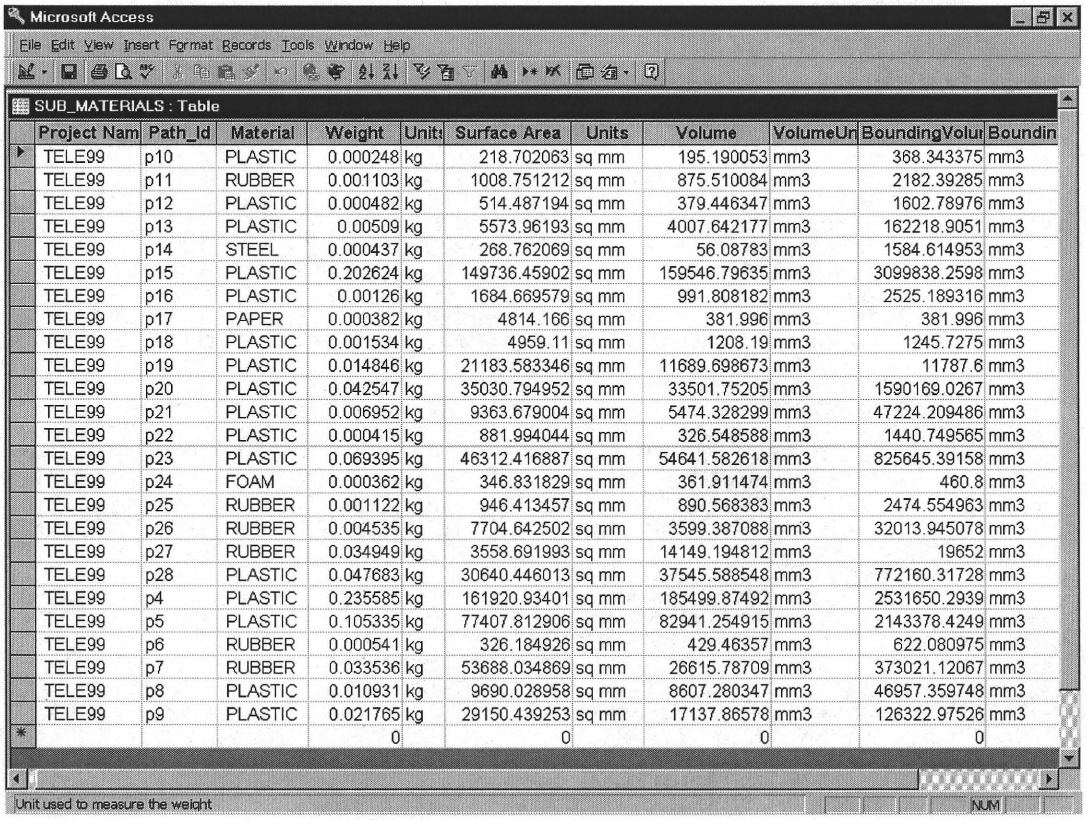
Record: 2 of 28

Name of Project

Fig. 5.9 Table PROJECT_ASSEMBLIES after population

When the program recursively traverses the product tree, it generates unique path identification for every component. With unique path identifications, MLCA can distinguish those components with same component name.

The population program uses Java Database Connectivity (JDBC) to connect to the MLCA database. According to MLCA database structure, the product design information is mainly stored to tables such as PROJECT, PROJECT-ASSEMBLIES, and SUB-MATERIALS. After running the database population program, the PROJECT table is updated and one new record is inserted in MLCA database for project as TELE99. The results for this TELE99 example are shown in Figs. 5.8-5.10.

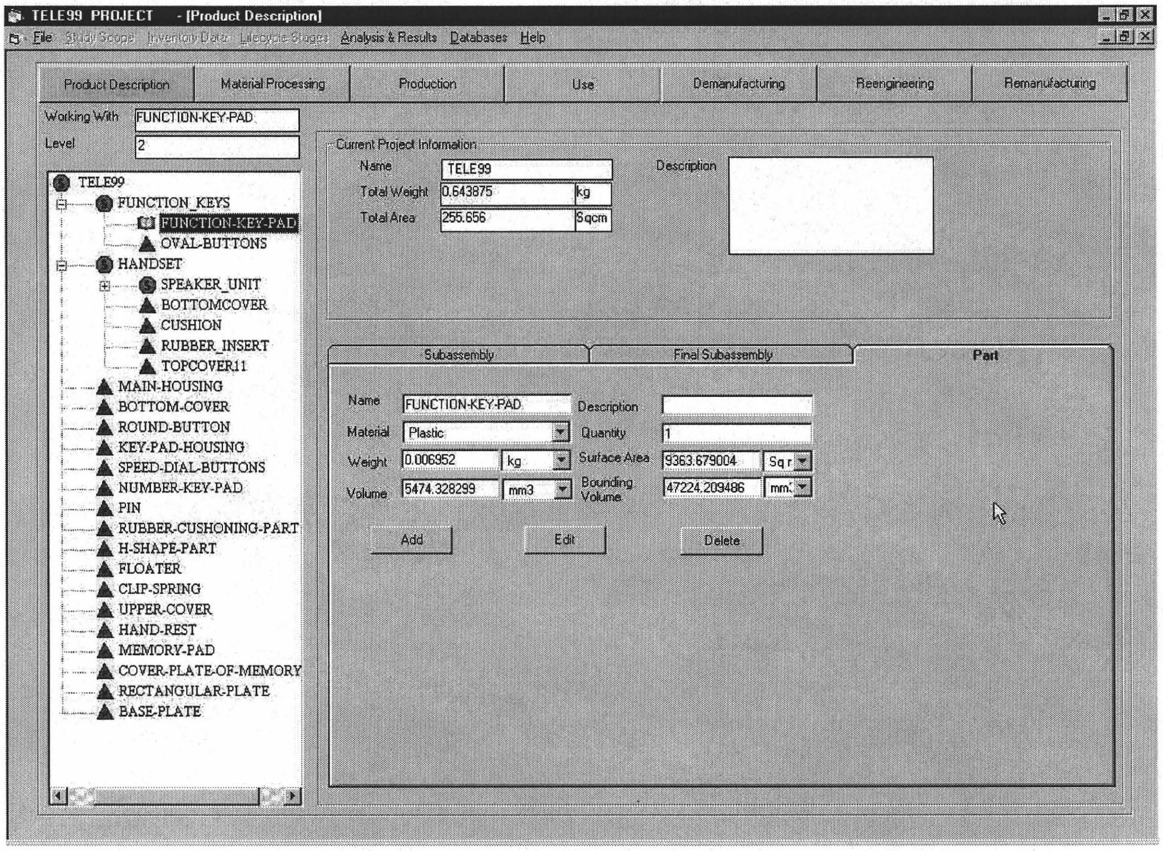


Project Nam	Path_Id	Material	Weight	Units	Surface Area	Units	Volume	VolumeUn	BoundingVolu	Boundin
TELE99	p10	PLASTIC	0.000248	kg	218.702063	sq mm	195.190053	mm3	368.343375	mm3
TELE99	p11	RUBBER	0.001103	kg	1008.751212	sq mm	875.510084	mm3	2182.39285	mm3
TELE99	p12	PLASTIC	0.000482	kg	514.487194	sq mm	379.446347	mm3	1602.78976	mm3
TELE99	p13	PLASTIC	0.00509	kg	5573.96193	sq mm	4007.642177	mm3	162218.9051	mm3
TELE99	p14	STEEL	0.000437	kg	268.762069	sq mm	56.08783	mm3	1584.614953	mm3
TELE99	p15	PLASTIC	0.202624	kg	149736.45902	sq mm	159546.79635	mm3	3099838.2598	mm3
TELE99	p16	PLASTIC	0.00126	kg	1684.669579	sq mm	991.808182	mm3	2525.189316	mm3
TELE99	p17	PAPER	0.000382	kg	4814.166	sq mm	381.996	mm3	381.996	mm3
TELE99	p18	PLASTIC	0.001534	kg	4959.11	sq mm	1208.19	mm3	1245.7275	mm3
TELE99	p19	PLASTIC	0.014846	kg	21183.583346	sq mm	11689.698673	mm3	11787.6	mm3
TELE99	p20	PLASTIC	0.042547	kg	35030.794952	sq mm	33501.75205	mm3	1590169.0267	mm3
TELE99	p21	PLASTIC	0.006952	kg	9363.679004	sq mm	5474.328299	mm3	47224.209486	mm3
TELE99	p22	PLASTIC	0.000415	kg	881.994044	sq mm	326.548588	mm3	1440.749565	mm3
TELE99	p23	PLASTIC	0.069395	kg	46312.416887	sq mm	54641.582618	mm3	825645.39158	mm3
TELE99	p24	FOAM	0.000362	kg	346.831829	sq mm	361.911474	mm3	460.8	mm3
TELE99	p25	RUBBER	0.001122	kg	946.413457	sq mm	890.568383	mm3	2474.554963	mm3
TELE99	p26	RUBBER	0.004535	kg	7704.642502	sq mm	3599.387088	mm3	32013.945078	mm3
TELE99	p27	RUBBER	0.034949	kg	3558.691993	sq mm	14149.194812	mm3	19652	mm3
TELE99	p28	PLASTIC	0.047683	kg	30640.446013	sq mm	37545.588548	mm3	772160.31728	mm3
TELE99	p4	PLASTIC	0.235585	kg	161920.93401	sq mm	185499.87492	mm3	2531650.2939	mm3
TELE99	p5	PLASTIC	0.105335	kg	77407.812906	sq mm	82941.254915	mm3	2143378.4249	mm3
TELE99	p6	RUBBER	0.000541	kg	326.184926	sq mm	429.46357	mm3	622.080975	mm3
TELE99	p7	RUBBER	0.033536	kg	53688.034869	sq mm	26615.78709	mm3	373021.12067	mm3
TELE99	p8	PLASTIC	0.010931	kg	9690.028958	sq mm	8607.280347	mm3	46957.359748	mm3
TELE99	p9	PLASTIC	0.021765	kg	29150.439253	sq mm	17137.86578	mm3	126322.97526	mm3
			0		0		0		0	

Unit used to measure the weight

Fig. 5.10 Table SUB_MATERIALS after population

Using the developed CAD-MLCA interface and database population program, users can see the tree structure of TELE99 and its related design information in Product Description stage of MLCA software as shown in Fig. 5.11. Thus the MLCA database is ready to perform the multi-lifecycle assessment and to study environmental impacts. Fig. 5.12 shows the environmental burdens emission of TELE99, which is one of the analysis results in MLCA.



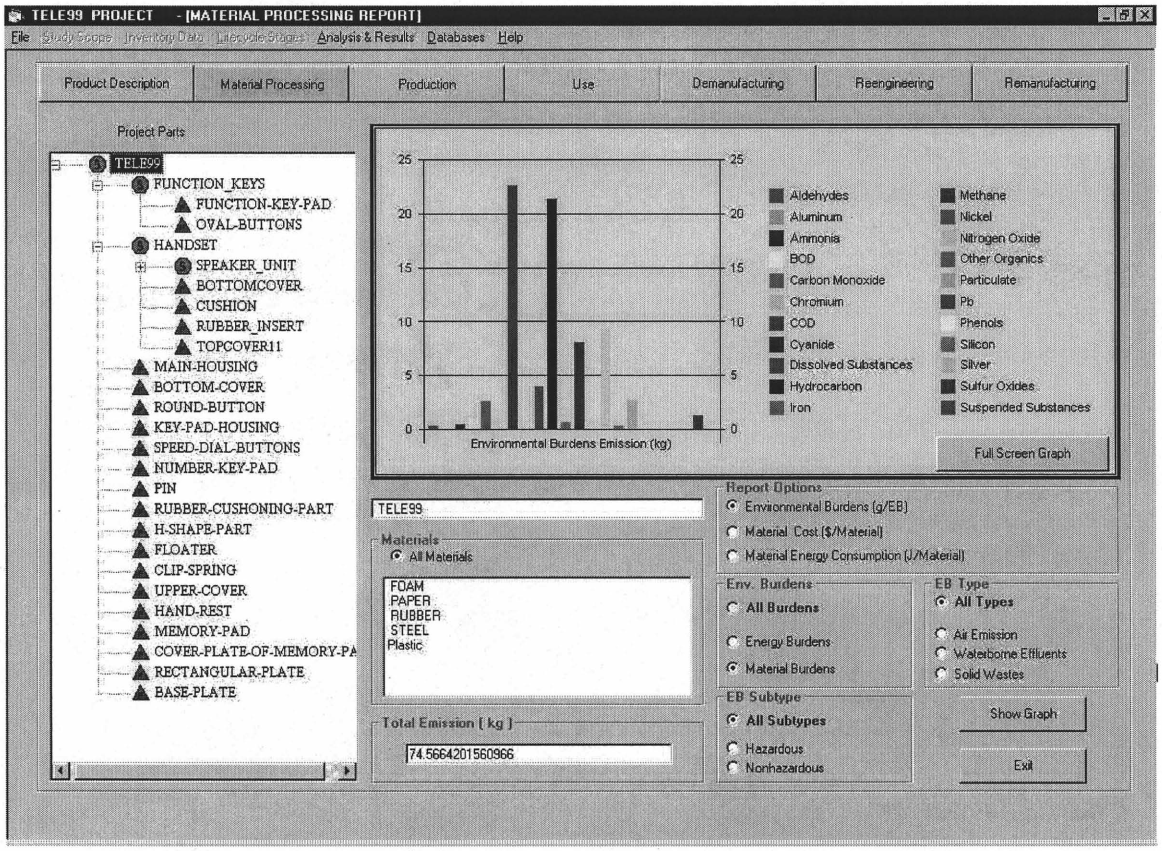
Working With: FUNCTION-KEY-PAD
Level: 2

```
TELE99
  FUNCTION_KEYS
    FUNCTION-KEY-PAD
    OVAL-BUTTONS
  HANDSET
    SPEAKER_UNIT
    BOTTOMCOVER
    CUSHION
    RUBBER_INSERT
    TOPCOVER11
  MAIN-HOUSING
  BOTTOM-COVER
  ROUND-BUTTON
  KEY-PAD-HOUSING
  SPEED-DIAL-BUTTONS
  NUMBER-KEY-PAD
  PIN
  RUBBER-CUSHONING-PART
  H-SHAPE-PART
  FLOATER
  CLIP-SPRING
  UPPER-COVER
  HAND-REST
  MEMORY-PAD
  COVER-PLATE-OF-MEMORY
  RECTANGULAR-PLATE
  BASE-PLATE
```

Current Project Information
Name: TELE99
Total Weight: 0.643875 kg
Total Area: 255.656 Sqcm
Description:

Part
Name: FUNCTION-KEY-PAD
Description:
Material: Plastic
Quantity: 1
Weight: 0.006952 kg
Surface Area: 9363.679004 Sq r
Volume: 5474.328299 mm3
Bounding Volume: 47224.209486 mm:

Fig. 11 Product description of TELE99 after population



TELE99
- FUNCTION_KEYS
  - FUNCTION-KEY-PAD
  - OVAL-BUTTONS
- HANDSET
  - SPEAKER_UNIT
  - BOTTOMCOVER
  - CUSHION
  - RUBBER_INSERT
  - TOPCOVER11
- MAIN-HOUSING
- BOTTOM-COVER
- ROUND-BUTTON
- KEY-PAD-HOUSING
- SPEED-DIAL-BUTTONS
- NUMBER-KEY-PAD
- PIN
- RUBBER-CUSHONING-PART
- H-SHAPE-PART
- FLOATER
- CLIP-SPRING
- UPPER-COVER
- HAND-REST
- MEMORY-PAD
- COVER-PLATE-OF-MEMORY-PA
- RECTANGULAR-PLATE
- BASE-PLATE

TELE99

Materials: [✓ All Materials]

FOAM
PAPER
RUBBER
STEEL
Plastic

Total Emission [kg]: 74.5664201560966

Report Options: [✓ Environmental Burdens (g/EB)] Material Cost ($/Material) Material Energy Consumption (J/Material)

Env. Burdens: All Burdens, Energy Burdens, [✓ Material Burdens]

EB Type: [✓ All Types] Air Emission, Waterborne Effluents, Solid Wastes

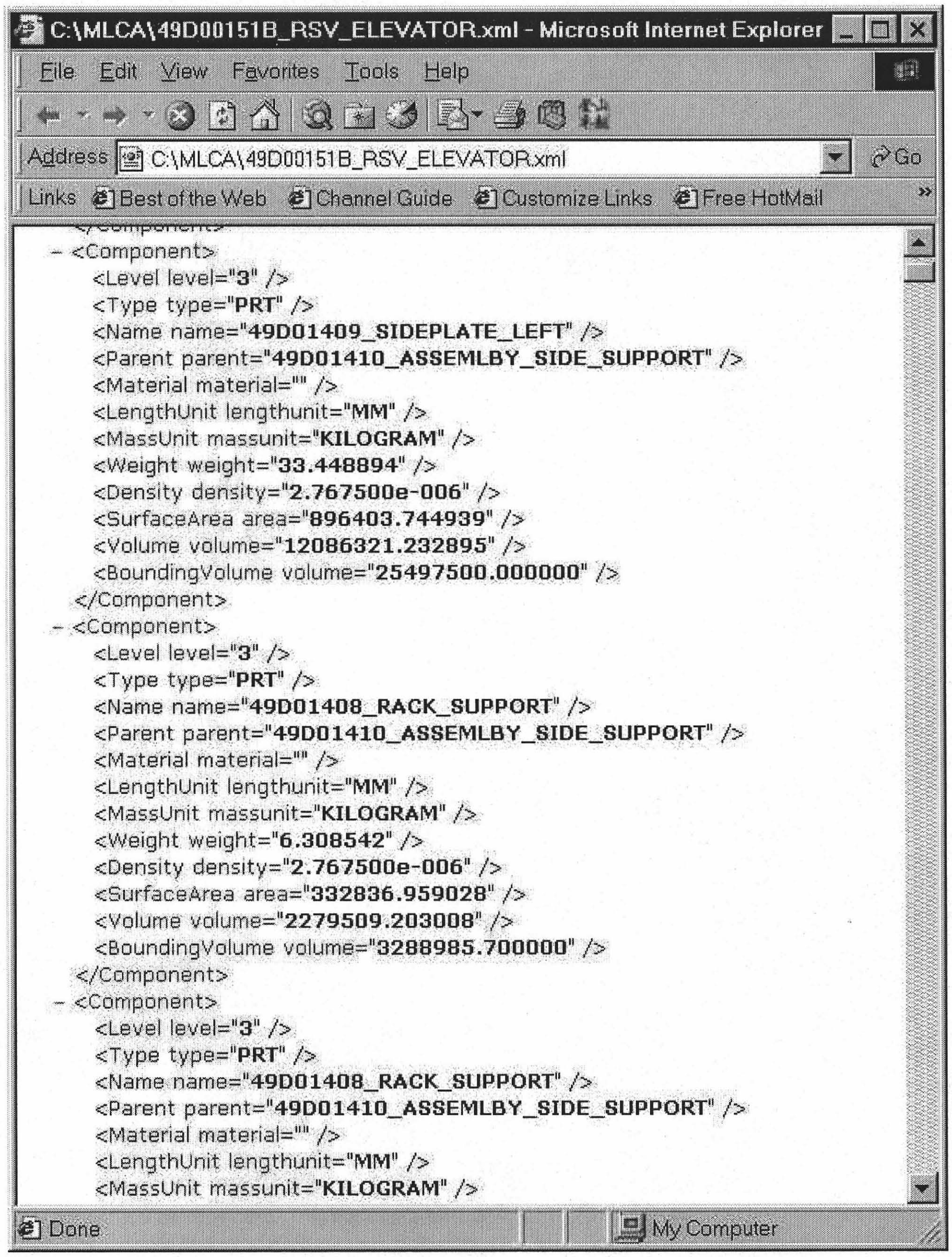
EB Subtype: [✓ All Subtypes] Hazardous, Nonhazardous

Fig. 12 Environmental burdens emission of TEL99

5.7 Case Study: RSV_ELEVATOR

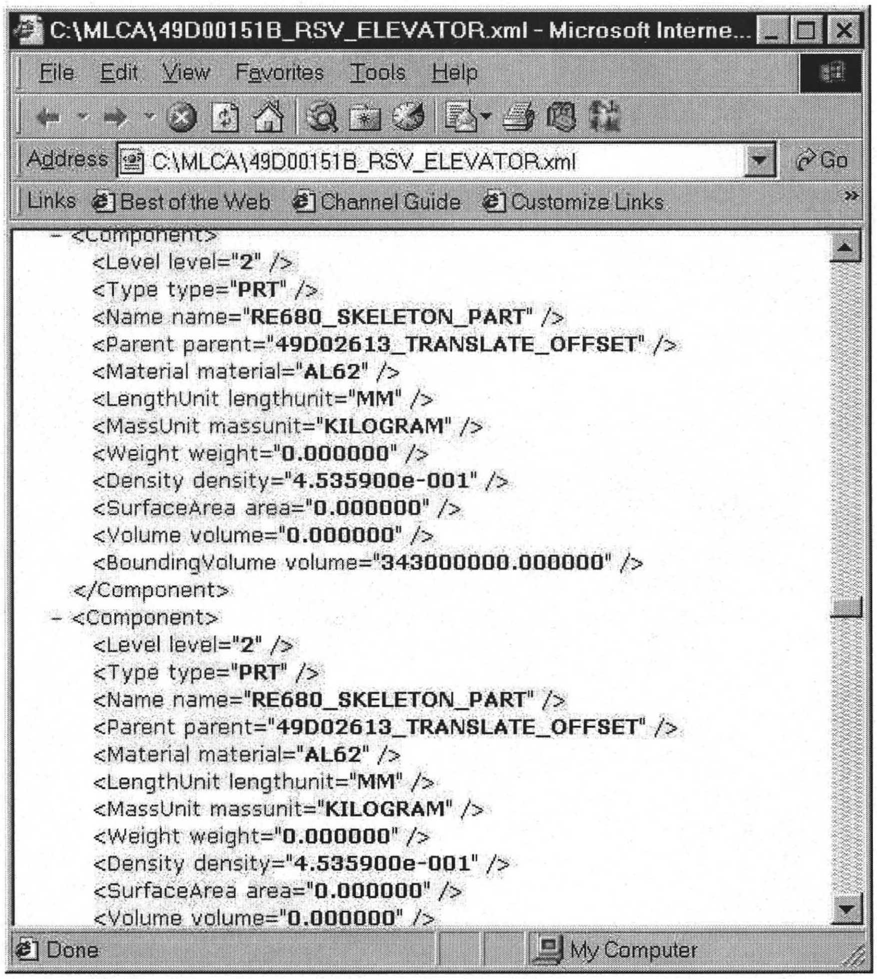
Normally, all the design information should be included in Pro/ENGINEER database if the design of a product is complete. But in some cases, some design information are missing from Pro/ENGINEER database, implying that something is wrong. For example, material or weight information does not exist for some components. To show how the CAD-MLCA interface to handle this kind situation, a product called 'RSV_ELEVATOR' is chosen as the sample.

In Fig. 5.13, we can find that there is no material specified for component '49D01409_SIDEPLATE_LEFT'. And weight information is missing for


C:\MLCA\49D00151B_RSV_ELEVATOR.xml - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address C:\MLCA\49D00151B_RSV_ELEVATOR.xml
Go
Links Best of the Web Channel Guide Customize Links Free HotMail
Done
My Computer

```
  - <Component>
      <Level level="3" />
      <Type type="PRT" />
      <Name name="49D01409_SIDEPLATE_LEFT" />
      <Parent parent="49D01410_ASSEMLBY_SIDE_SUPPORT" />
      <Material material="" />
      <LengthUnit lengthunit="MM" />
      <MassUnit massunit="KILOGRAM" />
      <Weight weight="33.448894" />
      <Density density="2.767500e-006" />
      <SurfaceArea area="896403.744939" />
      <Volume volume="12086321.232895" />
      <BoundingVolume volume="25497500.000000" />
    </Component>
  - <Component>
      <Level level="3" />
      <Type type="PRT" />
      <Name name="49D01408_RACK_SUPPORT" />
      <Parent parent="49D01410_ASSEMLBY_SIDE_SUPPORT" />
      <Material material="" />
      <LengthUnit lengthunit="MM" />
      <MassUnit massunit="KILOGRAM" />
      <Weight weight="6.308542" />
      <Density density="2.767500e-006" />
      <SurfaceArea area="332836.959028" />
      <Volume volume="2279509.203008" />
      <BoundingVolume volume="3288985.700000" />
    </Component>
  - <Component>
      <Level level="3" />
      <Type type="PRT" />
      <Name name="49D01408_RACK_SUPPORT" />
      <Parent parent="49D01410_ASSEMLBY_SIDE_SUPPORT" />
      <Material material="" />
      <LengthUnit lengthunit="MM" />
      <MassUnit massunit="KILOGRAM" />
```

Fig. 5.13 Components with no material


C:\MLCA\49D00151B_RSV_ELEVATOR.xml - Microsoft Interne...
File Edit View Favorites Tools Help
Address C:\MLCA\49D00151B_RSV_ELEVATOR.xml
Go
Links Best of the Web Channel Guide Customize Links
Done
My Computer

```
- <Component>
    <Level level="2" />
    <Type type="PRT" />
    <Name name="RE680_SKELETON_PART" />
    <Parent parent="49D02613_TRANSLATE_OFFSET" />
    <Material material="AL62" />
    <LengthUnit lengthunit="MM" />
    <MassUnit massunit="KILOGRAM" />
    <Weight weight="0.000000" />
    <Density density="4.535900e-001" />
    <SurfaceArea area="0.000000" />
    <Volume volume="0.000000" />
    <BoundingVolume volume="343000000.000000" />
  </Component>
- <Component>
    <Level level="2" />
    <Type type="PRT" />
    <Name name="RE680_SKELETON_PART" />
    <Parent parent="49D02613_TRANSLATE_OFFSET" />
    <Material material="AL62" />
    <LengthUnit lengthunit="MM" />
    <MassUnit massunit="KILOGRAM" />
    <Weight weight="0.000000" />
    <Density density="4.535900e-001" />
    <SurfaceArea area="0.000000" />
    <Volume volume="0.000000" />
```

Fig. 5.14 Component with no weight

'RE680_SKELETON_PART' in Fig. 5.14. In this case, population program will put "No material" or "No Weight" into those MLCA tables instead of just leaving attributes NULL or default value.

Once the population program realizes that there are some missing information for a product, it creates a 'ProE_Err' directory (in the case that the specified directory does not exist) and store all the components with error data items to the corresponding files, e.g., No_Weight.txt, No_Material.txt, and so on, which shown in Fig.5.15.

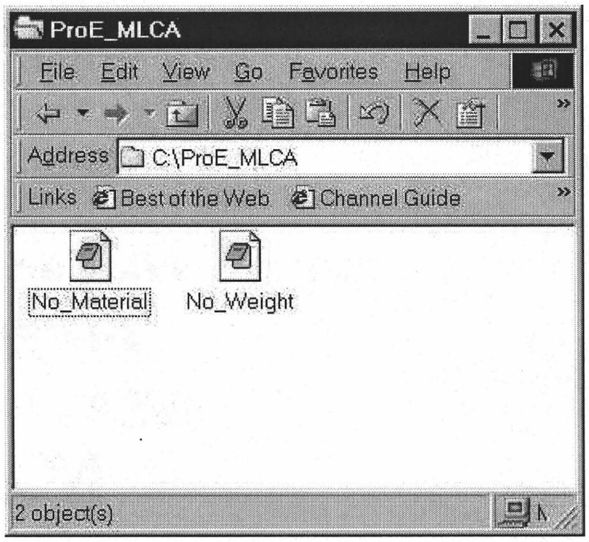
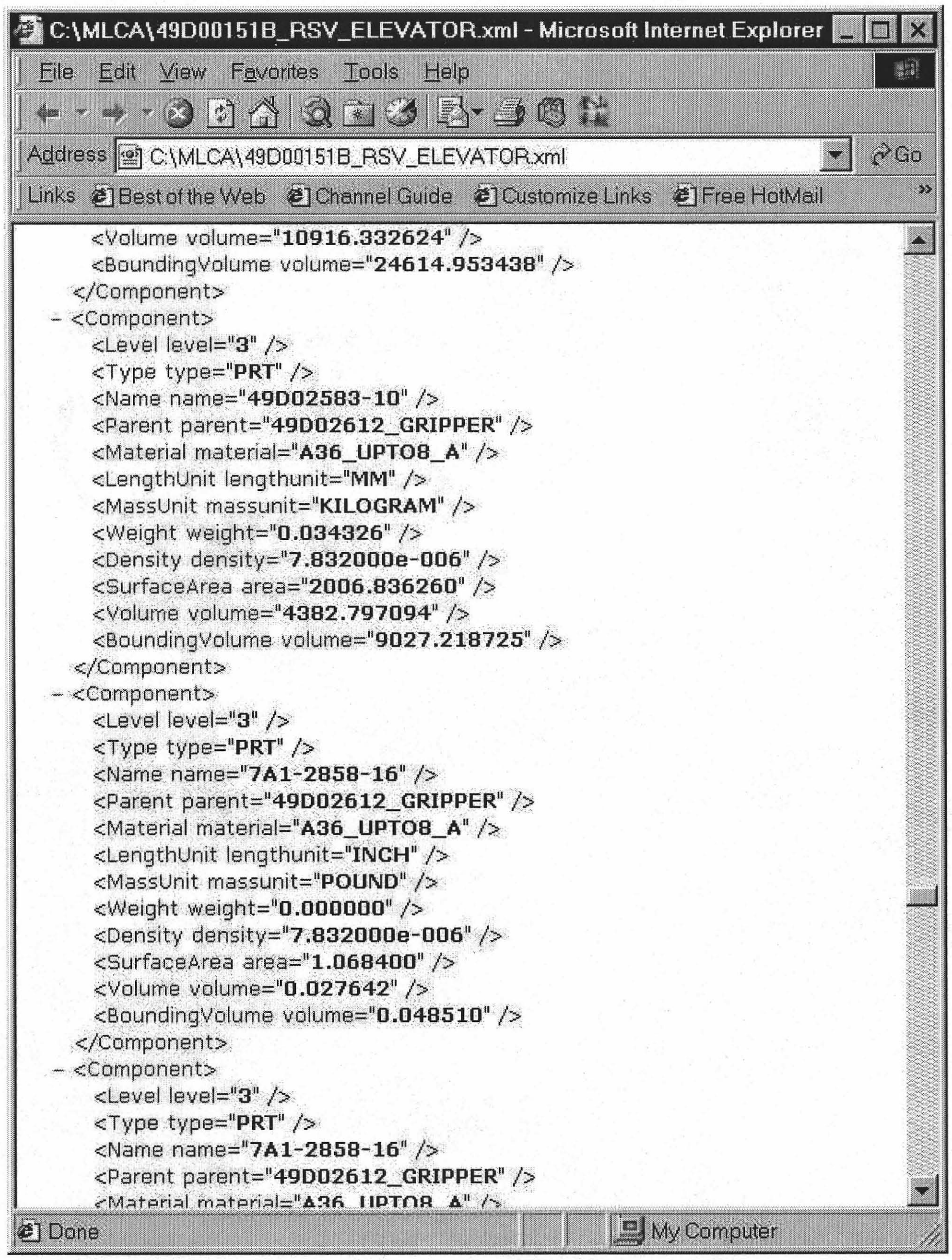

ProE_MLCA
File Edit View Go Favorites Help
Address C:\ProE_MLCA
Links Best of the Web Channel Guide
No_Material
No_Weight
2 object(s)

Fig. 5.15 Files containing error information stored in c:\ProE_MLCA

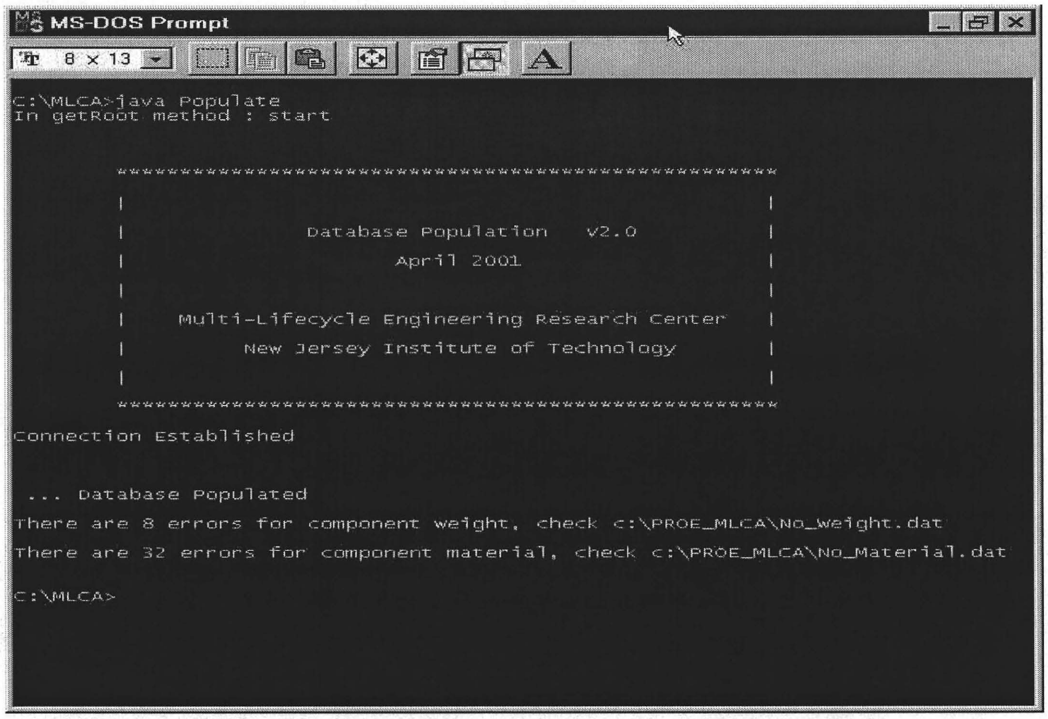
Another difference of this example product RSV_ELEVATOR from product TELE99 is that its component are designed by different designers, which use different sets of units. As shown in Fig. 5.16, the length unit and mass unit of component '49D02583-10' are millimeter and kilogram respectively, and for '7A1-2858-16' are inch and pound.

At the end of population, the program prompts a message, as shown in Fig. 5.17, to users about the error information. Thus they can easily locate the components and feedback to designers to correct them. Fig. 5.18 is the file about all the components without specified materials.


C:\MLCA\49D00151B_RSV_ELEVATOR.xml - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address C:\MLCA\49D00151B_RSV_ELEVATOR.xml
Go
Links Best of the Web Channel Guide Customize Links Free HotMail
Done
My Computer

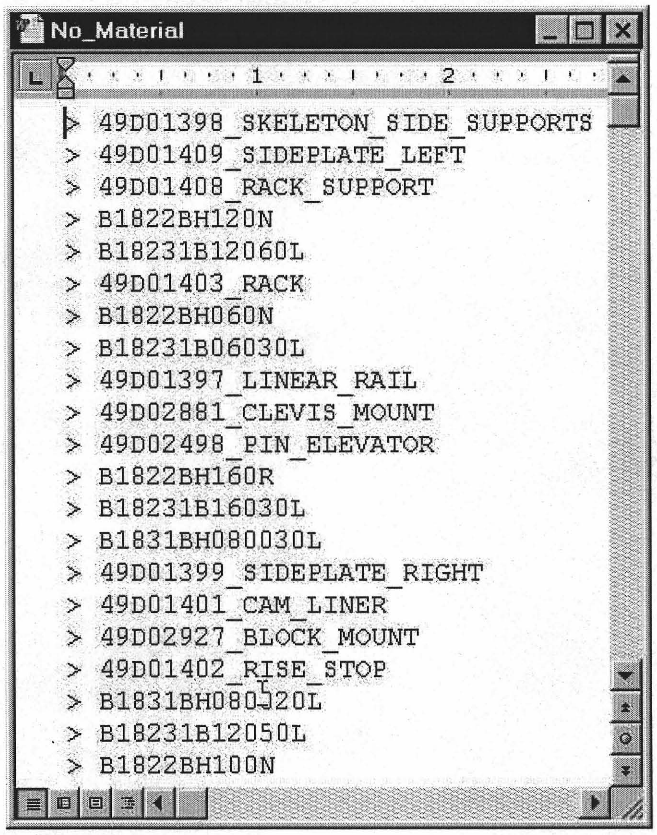
```
    <Volume volume="10916.332624" />
    <BoundingVolume volume="24614.953438" />
  </Component>
- <Component>
    <Level level="3" />
    <Type type="PRT" />
    <Name name="49D02583-10" />
    <Parent parent="49D02612_GRIPPER" />
    <Material material="A36_UPTO8_A" />
    <LengthUnit lengthunit="MM" />
    <MassUnit massunit="KILOGRAM" />
    <Weight weight="0.034326" />
    <Density density="7.832000e-006" />
    <SurfaceArea area="2006.836260" />
    <Volume volume="4382.797094" />
    <BoundingVolume volume="9027.218725" />
  </Component>
- <Component>
    <Level level="3" />
    <Type type="PRT" />
    <Name name="7A1-2858-16" />
    <Parent parent="49D02612_GRIPPER" />
    <Material material="A36_UPTO8_A" />
    <LengthUnit lengthunit="INCH" />
    <MassUnit massunit="POUND" />
    <Weight weight="0.000000" />
    <Density density="7.832000e-006" />
    <SurfaceArea area="1.068400" />
    <Volume volume="0.027642" />
    <BoundingVolume volume="0.048510" />
  </Component>
- <Component>
    <Level level="3" />
    <Type type="PRT" />
    <Name name="7A1-2858-16" />
    <Parent parent="49D02612_GRIPPER" />
    <Material material="A36_UPTO8_A" />
```

Fig. 5.16 Component with different sets of units



```
C:\MLCA>java Populate
In getRoot method : start

        ************************************************************
        |                                                          |
        |              Database Population    v2.0                 |
        |                    April 2001                            |
        |                                                          |
        |      Multi-Lifecycle Engineering Research Center         |
        |            New Jersey Institute of Technology            |
        |                                                          |
        ************************************************************
Connection Established

 ... Database Populated
There are 8 errors for component weight, check c:\PROE_MLCA\No_Weight.dat
There are 32 errors for component material, check c:\PROE_MLCA\No_Material.dat

C:\MLCA>
```

Fig. 5.17 Message prompted to designers



```
> 49D01398_SKELETON_SIDE_SUPPORTS
> 49D01409_SIDEPLATE_LEFT
> 49D01408_RACK_SUPPORT
> B1822BH120N
> B18231B12060L
> 49D01403_RACK
> B1822BH060N
> B18231B06030L
> 49D01397_LINEAR_RAIL
> 49D02881_CLEVIS_MOUNT
> 49D02498_PIN_ELEVATOR
> B1822BH160R
> B18231B16030L
> B1831BH080030L
> 49D01399_SIDEPLATE_RIGHT
> 49D01401_CAM_LINER
> 49D02927_BLOCK_MOUNT
> 49D01402_RISE_STOP
> B1831BH080020L
> B18231B12050L
> B1822BH100N
```

Fig. 5.18 File containing the components without specified materials

By using the CAD-MLCA interface, the design information of a product has been inserted in MLCA database in one mouse click. Thus, it reduces the manual work and also eliminates errors that may occur by manually entering the data, making MLCA software more user-friendly, less tedious and less error prone. Another benefit of the CAD-MLCA interface is that it provides the possibility for future research to feedback the MLCA analysis report directly to designers for judgement of better design. Thus, designers can improve their design of the product to reach better MLCA results. Users can keep this procedure on and on until a design meets the specified environmental requirements. This kind of closed-loop development systems are not only more accurate, but also cost-saving and time efficient. The presented work can thus help significantly put DFE concepts into the industrial use.

CHAPTER 6

SUMMARY AND FUTURE RESEARCH

6.1 Contributions of This Thesis

The CAD-MLCA interface acts as a backbone for the next generation DFE tool. The presented CAD-MLCA interface can effectively generate the required data from product CAD files and populate them into the proposed MLCA data. This implies a significant saving in data entry and prevention from error entry.

The work represents an important step for designers to perform rapid and timely multi-life cycle assessment of a product and receive the environmental results as feedback for better design. This will foster the wide application of DFE into not only commercial products, but also military systems.

6.2 Limitations and Future Research

Data collection remains a main issue for the lifecycle assessment and their associative databases. To perform an accurate multi-lifecycle assessment on a product, data and information are needed on energy consumption, materials usage, and environmental burdens of each process in the entire product life cycle in addition to the vast amount of information on a product from its CAD files. While several existing databases are already of great help, more future research is needed, especially, for example, demanufacturing processes.

Integration of CAD software and MLCA software is the key issue for the next generation DFE tools. The presented CAD-MLCA interface not only makes it possible, but also improves the degree of integration significantly. However, when users take a closer look at the whole structure of the interface and MLCA software itself, the following problems should be addressed in the future research:

■ Integration

As shown in Fig. 6.1, too many programming languages are involved in the interface, i.e., C in CAD tool, Java in population program, and VB in MLCA DBMS. This leaves much room for a better integration by using a single language. Actually, as a utility part, the population program should be embedded into MLCA DBMS. The present use of JAVA for the interface and VB for MLCA makes this very different if not impossible.

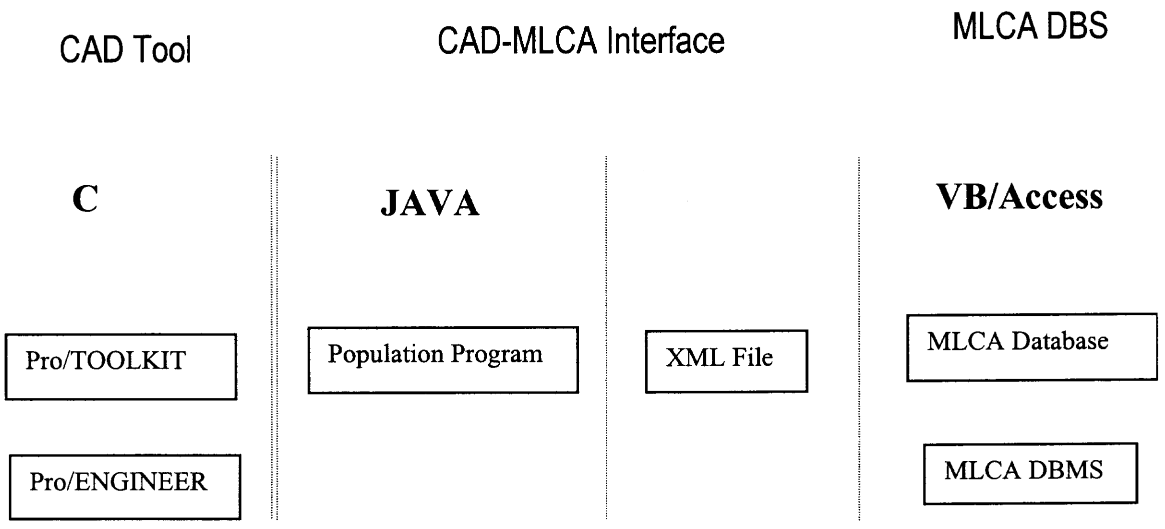


Fig. 6.1 The structure inside the DFE tools

■ Security

Although Microsoft Access database is easy to use, and Visual Basic also provides user-friendly interface for software development, the VB/Access mode can't satisfy many requirements for large scale DBS like MLCA software. For example, database security is a serious issue for Access database. Due to so many different kinds of users mentioned in Chapter 3, MLCA software should provide variable levels of corresponding privileges for different users to access to the database. Unfortunately, Microsoft Access database can not handle this issue in a convenient way.

■ Speed

The key weakness of Visual Basic is that it has no *pointer*, which means that VB can't manipulate hardware (memory addresses) directly. When the number of data items becomes significant large, the running speed of the program is expected to decrease dramatically.

For new version of MLCA software, ORACLE has significant advantages. ORACLE employs a robust security mechanism for database management. It also provides its embedded C or JAVA to develop software. Both C and JAVA can make use of *pointer* to directly manipulate memory addresses. Thus, it can not only keep a higher running speed, but also improve the integration significantly. Fig. 6.2 shows the structure of the new version of DFE tools.

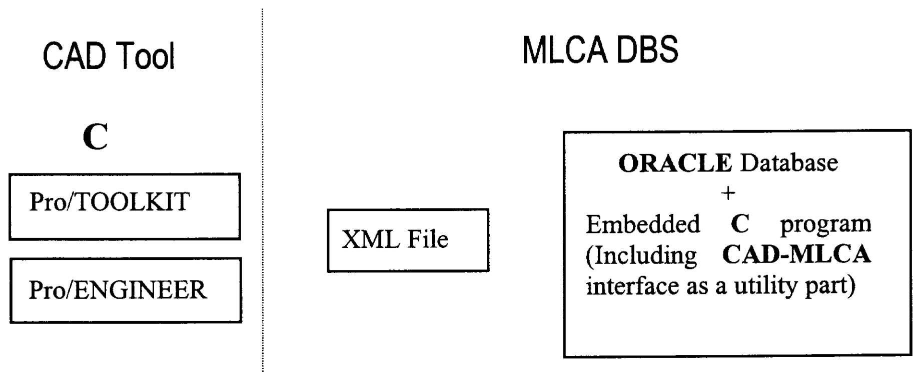

CAD Tool
C
Pro/TOOLKIT
Pro/ENGINEER
MLCA DBS
XML File
ORACLE Database + Embedded C program (Including CAD-MLCA interface as a utility part)

Fig. 6.2 Structure of new version DFE tools

APPENDIX

TELE99.XML

```
<?xml version="1.0"?>
<start><Component> <Level level= " 0" /> <Type type = " ASM" /> <Name name= "
TELE99" /> <Parent parent= "NULL" />      </Component>

<Component> <Level level = "1" /> <Type type = " PRT " /> <Name name = "MAIN-
HOUSING" /> <Parent parent= " TELE99" /> <Material material = " PLASTIC " />
<LengthUnit lengthunit=" MM "/> <MassUnit massunit=" KILOGRAM " /> <Weight
weight=" 0.235585 " /> <Density density=" 1.270000e-006 " /> <SurfaceArea area="
161920.934009 " />  <Volume volume=" 185499.874916 " /> <BoundingVolume
boundingvolume=" 2531650.293899 " /> </Component>

<Component> <Level level = "1" /> <Type type = " PRT " /> <Name name = "BOTTOM-
COVER" />   <Parent parent= " TELE99" /> <Material material = " PLASTIC " />
<LengthUnit lengthunit=" MM "/> <MassUnit massunit=" KILOGRAM " /> <Weight
weight=" 0.105335 " /> <Density density=" 1.270000e-006 " /> <SurfaceArea area="
77407.812906 " /> <Volume volume=" 82941.254915 " /> <BoundingVolume
boundingvolume=" 2143378.424915 " /> </Component>

<Component> <Level level = "1" /> <Type type = " PRT " /> <Name name = "ROUND-
BUTTON" />  <Parent parent= " TELE99" /> <Material material = " RUBBER " />
<LengthUnit lengthunit=" MM "/> <MassUnit massunit=" KILOGRAM " /> <Weight
weight=" 0.000541 " /> <Density density=" 1.260000e-006 " />  <SurfaceArea
area=" 326.184926 " /> <Volume volume=" 429.463570 " /> <BoundingVolume
boundingvolume="     622.080975 " /> </Component>

<Component> <Level level = "1" /> <Type type = " PRT " /> <Name name = "ROUND-
BUTTON" />  <Parent parent= " TELE99" /> <Material material = " RUBBER " />
<LengthUnit lengthunit=" MM "/> <MassUnit massunit=" KILOGRAM " /> <Weight
weight=" 0.000541 " /> <Density density=" 1.260000e-006 " /> <SurfaceArea area="
326.184926 " /> <Volume volume=" 429.463570 " /> <BoundingVolume
boundingvolume="     622.080975 " /> </Component>

<Component> <Level level = "1" /> <Type type = " PRT " /> <Name name = "ROUND-
BUTTON" />  <Parent parent= " TELE99" /> <Material material = " RUBBER " />
<LengthUnit lengthunit=" MM "/>   <MassUnit massunit=" KILOGRAM " /> <Weight
weight=" 0.000541 " /> <Density density=" 1.260000e-006 " /> <SurfaceArea area="
326.184926 " /> <Volume volume=" 429.463570 " /> <BoundingVolume
boundingvolume="     622.080975 " /> </Component>

<Component> <Level level = "1" /> <Type type = " PRT " /> <Name name = "ROUND-
BUTTON" />  <Parent parent= " TELE99" /> <Material material = " RUBBER " />
<LengthUnit lengthunit=" MM "/> <MassUnit massunit=" KILOGRAM " /> <Weight
weight=" 0.000541 " /> <Density density=" 1.260000e-006 " /> <SurfaceArea area="
326.184926 " /> <Volume volume=" 429.463570 " /> <BoundingVolume
boundingvolume="     622.080975 " /> </Component>

<Component> <Level level = "1" /> <Type type = " PRT " /> <Name name = "KEY-PAD-
HOUSING" /> <Parent parent= " TELE99" /> <Material material = " RUBBER " />
<LengthUnit lengthunit=" MM "/> <MassUnit massunit=" KILOGRAM " /> <Weight
weight=" 0.033536 " /> <Density density=" 1.260000e-006 " /> <SurfaceArea area="
```

53688.034869 " /> <Volume volume=" 26615.787090 " /> <BoundingVolume boundingvolume=" 373021.120670 " /> </Component>

<Component> <Level level = "1" /> <Type type = " PRT " /> <Name name = "SPEED-DIAL-BUTTONS" /> <Parent parent= " TELE99" /> <Material material = " PLASTIC " /> <LengthUnit lengthunit=" MM "/> <MassUnit massunit=" KILOGRAM " /> <Weight weight=" 0.010931 " /> <Density density=" 1.270000e-006 " /> <SurfaceArea area=" 9690.028958 " /> <Volume volume=" 8607.280347 " /> <BoundingVolume boundingvolume="   46957.359748 " /> </Component>

<Component> <Level level = "1" /> <Type type = " PRT " /> <Name name = "NUMBER-KEY-PAD" /> <Parent parent= " TELE99" /> <Material material = " PLASTIC " /> <LengthUnit lengthunit=" MM "/> <MassUnit massunit=" KILOGRAM " /> <Weight weight=" 0.021765 " /> <Density density=" 1.270000e-006 " /> <SurfaceArea area=" 29150.439253 " /> <Volume volume=" 17137.865780 " /> <BoundingVolume boundingvolume=" 126322.975263 " /> </Component>

<Component> <Level level = "1" /> <Type type = " PRT " /> <Name name = "PIN" /> <Parent parent= " TELE99" /> <Material material = " PLASTIC " /> <LengthUnit lengthunit=" MM "/> <MassUnit massunit=" KILOGRAM " /> <Weight weight=" 0.000248 " /> <Density density=" 1.270000e-006 " /> <SurfaceArea area=" 218.702063 " /> <Volume volume=" 195.190053 " /> <BoundingVolume boundingvolume="   368.343375 " /> </Component>

<Component> <Level level = "1" /> <Type type = " PRT " /> <Name name = "RUBBER-CUSHONING-PART" /> <Parent parent= " TELE99" /> <Material material = " RUBBER " /> <LengthUnit lengthunit=" MM "/> <MassUnit massunit=" KILOGRAM " />   <Weight weight=" 0.001103 " /> <Density density=" 1.260000e-006 " /> <SurfaceArea area=" 1008.751212 " /> <Volume volume=" 875.510084 " /> <BoundingVolume boundingvolume="   2182.392850 " /> </Component>

<Component> <Level level = "1" /> <Type type = " PRT " /> <Name name = "H-SHAPE-PART" /> <Parent parent= " TELE99" /> <Material material = " PLASTIC " /> <LengthUnit lengthunit=" MM "/> <MassUnit massunit=" KILOGRAM " /> <Weight weight=" 0.000482 " /> <Density density=" 1.270000e-006 " /> <SurfaceArea area=" 514.487194 " /> <Volume volume=" 379.446347 " /> <BoundingVolume boundingvolume="   1602.789760 " /> </Component>

<Component> <Level level = "1" /> <Type type = " PRT " /> <Name name = "FLOATER" /> <Parent parent= " TELE99" /> <Material material = " PLASTIC " /> <LengthUnit lengthunit=" MM "/> <MassUnit massunit=" KILOGRAM " /> <Weight weight=" 0.005090 " /> <Density density=" 1.270000e-006 " /> <SurfaceArea area=" 5573.961930 " /> <Volume volume=" 4007.642177 " /> <BoundingVolume boundingvolume=" 162218.905099 " /> </Component>

<Component> <Level level = "1" /> <Type type = " PRT " /> <Name name = "CLIP-SPRING" /> <Parent parent= " TELE99" /> <Material material = " STEEL " /> <LengthUnit lengthunit=" MM "/> <MassUnit massunit=" KILOGRAM " /> <Weight weight=" 0.000437 " /> <Density density=" 7.800000e-006 " /> <SurfaceArea area=" 268.762069 " /> <Volume volume=" 56.087830 " /> <BoundingVolume boundingvolume=" 1584.614953 " /> </Component>

<Component> <Level level = "1" /> <Type type = " PRT " /> <Name name = "UPPER-COVER" /> <Parent parent= " TELE99" /> <Material material = " PLASTIC " /> <LengthUnit lengthunit=" MM "/> <MassUnit massunit=" KILOGRAM " /> <Weight weight=" 0.202624 " /> <Density density=" 1.270000e-006 " /> <SurfaceArea area="

149736.459019 " /> <Volume volume=" 159546.796348 " /> <BoundingVolume boundingvolume=" 3099838.259814 " /> </Component>

<Component> <Level level = "1" /> <Type type = " PRT " /> <Name name = "HAND-REST" /> <Parent parent= " TELE99" /> <Material material = " PLASTIC " /> <LengthUnit lengthunit=" MM "/> <MassUnit massunit=" KILOGRAM " /> <Weight weight=" 0.001260 " /> <Density density=" 1.270000e-006 " /> <SurfaceArea area=" 1684.669579 " /> <Volume volume=" 991.808182 " /> <BoundingVolume boundingvolume=" 2525.189316 " /> </Component>

<Component> <Level level = "1" /> <Type type = " PRT " /> <Name name = "MEMORY-PAD" /> <Parent parent= " TELE99" /> <Material material = " PAPER " /> <LengthUnit lengthunit=" MM "/> <MassUnit massunit=" KILOGRAM " /> <Weight weight=" 0.000382 " /> <Density density=" 1.000000e-006 " /> <SurfaceArea area=" 4814.166000 " /> <Volume volume=" 381.996000 " /> <BoundingVolume boundingvolume=" 381.996000 " /> </Component>

<Component> <Level level = "1" /> <Type type = " PRT " /> <Name name = "COVER-PLATE-OF-MEMORY-PAD" /> <Parent parent= " TELE99" /> <Material material = " PLASTIC " /> <LengthUnit lengthunit=" MM "/> <MassUnit massunit=" KILOGRAM " /> <Weight weight=" 0.001534 " /> <Density density=" 1.270000e-006 " /> <SurfaceArea area=" 4959.110000 " /> <Volume volume=" 1208.190000 " /> <BoundingVolume boundingvolume=" 1245.727500 " /> </Component>

<Component> <Level level = "1" /> <Type type = " PRT " /> <Name name = "RECTANGULAR-PLATE" /> <Parent parent= " TELE99" /> <Material material = " PLASTIC " /> <LengthUnit lengthunit=" MM "/> <MassUnit massunit=" KILOGRAM " /> <Weight weight=" 0.014846 " /> <Density density=" 1.270000e-006 " /> <SurfaceArea area=" 21183.583346 " /> <Volume volume=" 11689.698673 " /> <BoundingVolume boundingvolume=" 11787.600000 " /> </Component>

<Component> <Level level = "1" /> <Type type = " PRT " /> <Name name = "BASE-PLATE" /> <Parent parent= " TELE99" /> <Material material = " PLASTIC " /> <LengthUnit lengthunit=" MM "/> <MassUnit massunit=" KILOGRAM " /> <Weight weight=" 0.042547 " /> <Density density=" 1.270000e-006 " /> <SurfaceArea area=" 35030.794952 " /> <Volume volume=" 33501.752050 " /> <BoundingVolume boundingvolume=" 1590169.026724 " /> </Component>

<Component> <Level level = "1" /> <Type type = " PRT " /> <Name name = "BASE-PLATE" /> <Parent parent= " TELE99" /> <Material material = " PLASTIC " /> <LengthUnit lengthunit=" MM "/> <MassUnit massunit=" KILOGRAM " /> <Weight weight=" 0.042547 " /> <Density density=" 1.270000e-006 " /> <SurfaceArea area=" 35030.794952 " /> <Volume volume=" 33501.752050 " /> <BoundingVolume boundingvolume=" 1590169.026724 " /> </Component>

<Component> <Level level = "1" /> <Type type = " ASM " /> <Name name = "FUNCTION_KEYS" /> <Parent parent= " TELE99" /></Component>

<Component> <Level level = "2" /> <Type type = " PRT " /> <Name name = "FUNCTION-KEY-PAD" /> <Parent parent= " FUNCTION_KEYS" /> <Material material = " PLASTIC " /> <LengthUnit lengthunit=" MM "/> <MassUnit massunit=" KILOGRAM " /> <Weight weight=" 0.006952 " /> <Density density=" 1.270000e-006 " /> <SurfaceArea area=" 9363.679004 " /> <Volume volume=" 5474.328299 " /> <BoundingVolume boundingvolume=" 47224.209486 " /> </Component>

<Component> <Level level = "2" /> <Type type = " PRT " /> <Name name = "OVAL-BUTTONS" /> <Parent parent= " FUNCTION_KEYS" /> <Material material = " PLASTIC "

/> <LengthUnit lengthunit=" MM "/> <MassUnit massunit=" KILOGRAM " /> <Weight weight=" 0.000415 " /> <Density density=" 1.270000e-006 " /> <SurfaceArea area=" 881.994044 " /> <Volume volume=" 326.548588 " /> <BoundingVolume boundingvolume="   1440.749565 " /> </Component>

<Component> <Level level = "1" /> <Type type = " ASM " /> <Name name = "HANDSET" /> <Parent parent= " TELE99" /> </Component>

<Component> <Level level = "2" /> <Type type = " PRT " /> <Name name = "BOTTOMCOVER" /> <Parent parent= " HANDSET" /> <Material material = " PLASTIC " /> <LengthUnit lengthunit=" MM "/> <MassUnit massunit=" KILOGRAM " /> <Weight weight=" 0.069395 " /> <Density density=" 1.270000e-006 " /> <SurfaceArea area=" 46312.416887 " /> <Volume volume=" 54641.582618 " /> <BoundingVolume boundingvolume=" 825645.391577 " /> </Component>

<Component> <Level level = "2" /> <Type type = " PRT " /> <Name name = "CUSHION" /> <Parent parent= " HANDSET" /> <Material material = " FOAM " /> <LengthUnit lengthunit=" MM "/> <MassUnit massunit=" KILOGRAM " /> <Weight weight=" 0.000362 " /> <Density density=" 1.000000e-006 " /> <SurfaceArea area=" 346.831829 " /> <Volume volume=" 361.911474 " /> <BoundingVolume boundingvolume="   460.800000 " /> </Component>

<Component> <Level level = "2" /> <Type type = " PRT " /> <Name name = "RUBBER_INSERT" /> <Parent parent= " HANDSET" /> <Material material = " RUBBER " /> <LengthUnit lengthunit=" MM "/> <MassUnit massunit=" KILOGRAM " /> <Weight weight=" 0.001122 " /> <Density density=" 1.260000e-006 " /> <SurfaceArea area=" 946.413457 " /> <Volume volume=" 890.568383 " /> <BoundingVolume boundingvolume="   2474.554963 " /> </Component>

<Component> <Level level = "2" /> <Type type = " ASM " /> <Name name = "SPEAKER_UNIT" /> <Parent parent= " HANDSET" /></Component>

<Component> <Level level = "3" /> <Type type = " PRT " /> <Name name = "SPEAKER_RUBBER_PROTECTION_" /> <Parent parent= " SPEAKER_UNIT" /> <Material material = " RUBBER " /> <LengthUnit lengthunit=" MM "/> <MassUnit massunit=" KILOGRAM " /> <Weight weight=" 0.004535 " /> <Density density=" 1.260000e-006 " /> <SurfaceArea area=" 7704.642502 " /> <Volume volume=" 3599.387088 " /> <BoundingVolume boundingvolume="   32013.945078 " /> </Component>

<Component> <Level level = "3" /> <Type type = " PRT " /> <Name name = "SPEAKER_" /> <Parent parent= " SPEAKER_UNIT" /> <Material material = " RUBBER " /> <LengthUnit lengthunit=" MM "/> <MassUnit massunit=" KILOGRAM " />   <Weight weight=" 0.034949 " /> <Density density=" 2.470000e-006 " /> <SurfaceArea area=" 3558.691993 " /> <Volume volume=" 14149.194812 " /> <BoundingVolume boundingvolume="   19652.000000 " /> </Component>

<Component> <Level level = "2" /> <Type type = " PRT " /> <Name name = "TOPCOVER11" /> <Parent parent= " HANDSET" /> <Material material = " PLASTIC " /> <LengthUnit lengthunit=" MM "/> <MassUnit massunit=" KILOGRAM " /> <Weight weight=" 0.047683 " /> <Density density=" 1.270000e-006 " /> <SurfaceArea area=" 30640.446013 " /> <Volume volume=" 37545.588548 " /> <BoundingVolume boundingvolume=" 772160.317276 " /> </Component>

</start>

REFERENCES

1. H. F. Al-Okush, *Design for Environment of Office Telephone and Electronic Products*, M.S. Thesis, Mechanical Engineering Dept., NJIT, NJ January 1999.

2. D. J. Badwe, *Multi-Lifecycle Assessment of Cathode Ray Tubes*, M.S. Thesis, Manufacturing System Engineering, NJIT, 1997.

3. R. J. Caudill, *A Final Proposal to The National Science Foundation Engineering Research Centers Program*, Multi-lifecycle Engineering and Manufacturing Program Proposal, Multi-Lifecycle Engineering Research Center, NJIT, NJ 1997.

4. R. J. Caudill et al, "Multi-Lifecycle Product Recovery for Electronic Products," *Multi-lifecycle Engineering and Manufacturing Program -Final Report 97,* Multi-Lifecycle Engineering Research Center, NJIT, NJ 1997.

5. R. J. Caudill et al, "Lifecycle Analysis of CRTs," *Multi-lifecycle Engineering and Manufacturing Program - Final Report 99*, Multi-Lifecycle Engineering Research Center, NJ 1999.

6. P. P. Chen, "Entity-Relationship Model -Towards a Unified View of Data," *ACM Transactions on Database System*, Vol.1, No.1, pp. 9-36, March 1976.

7. M. A. Curran, *Environmental Lifecycle Assessment*, McGraw-Hill, New York 1996.

8. R. Elmasri and S. B. Navathe, *Fundamentals Of Database System*, Addison-Wesley Publishing Company, 1994, ISBN 0-8053-1748-1.

9. Fava, et. al., *A Conceptal Framework for Life-Cycle Impact Assessment*, SETC 1992, Pensacola, FL.

10. J. Fiksel, "Design for Environment: The New Quality Imperative," *Corporate Environmental Strategy*, Vol. 1, No. 3, December 1993.

11. J. Fiksel, *Conceptual Principles of DFE, Design for Environment*, McGraw-Hill, New York, NY, 1996.

12. T. E. Graedel, *Streamlined Life-Cycle Assessment*, Prentice Hall, Upper Saddle River, New Jersey,1998, ISBN 0-13-607425-1.

13. T. E. Graedel, B. R. Allenby, *Design for Environment*, Prentice Hall, Upper Saddle River, NJ 1996, ISBN 0-13-531682-0.

14. J. Jin, *Multi-lifecycle Assessment Design Tools and Software Development*, M.S. Thesis, Computer Engineering, New Jersey Institute of Technology, Newark, NJ, January 1999.

15. A. K. Gregory et al, *Product Life Cycle Assessment to Reduce Health Risks and Environmental Impacts*, Noyes Data Corporation, Park Ridge, New Jersey 1994, ISBN 0-8155-1354-2.

16. G. Lausen and G. Vossen, *Models and Languages of Object-Oriented Database*, Addison Wesley Longman Ltd 1998, ISBN 0-201-62431-1.

17. M. Menke, Gary A. Davis and B. W. Vigon, *Evaluation of Life-Cycle Assessment Tools*, Center for Clean Products and Clean Technologies, the University of Tennessee, August 1996.

18. F. Pascal, *Understanding Relational Databases with Examples in SQL-92*, John Wiley & Sons. Inc, 1993 ISBN 0-471-58538-6.

19. PTC, Parametric Technology Corporations, *Pro/Toolkit Reference Guide,1999.*

20. Eco-it, PRé Consultants BV, "Eco-it: Eco-Indicator Tool for Environmentally Friendly Design," http://www.pre.nl/eco-it.html, March 2001.

21. SETAC Foundation, A Technical Framework for Life-cycle Assessments, SETAC, Pensacola, January 1991.

22. Ecobalance, Inc., "Ecobilan Group's Life-Cycle Assessment: TEAM," http://www.ecobilan.com/, March 2001

23. SimaPro, "SimaPro 4.0 - the Life Cycle Assessment Tool", Http://www.pre.nl/simapro.html, March 2001.

24. B. Suratran, *Integration of Multi-lifecycle Assessment and Design for Environment Database Using Relational Model Concepts*, M.S. Thesis, Computer Science, NJIT, Newark, NJ, January 2000.

25. Y. Tang, M. C. Zhou, and R. J. Caudill, "Dissembly Model, Planning, and Application: A Review," *Proc. of IEEE Int. Conf. on Robotics and Automation*, pp. 2197-2202, San Francisco, CA, April 2000.

26. M. C. Zhou, R. J. Caudill et al., "Multi-lifecycle Product Recovery for Electronic Product," *Electronics Manufacturing*, 9(1), 1015, March 1999.

27. E. Zussman and M. C. Zhou, "Design and Implementation of an Adaptive Planner for Disassembly Processes," *IEEE Trans. on Robotics and Automation*, Vol. 16, No. 2, pp. 171-179, April 2000.