WEB Based Cost Estimation Models for the Manufacturing of Advanced Composites

by

Joshua W. Pas

B.S., Industrial Engineering (1999)

Purdue University

Submitted to the Department of Mechanical Engineering
in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Mechanical Engineering

at the

Massachusetts Institute of Technology

June 2001

Signature of Author:_______

Department of Mechanical Engineering
May 11, 2001

Certified by:_______

Timothy Gutowski
Professor of Mechanical Engineering
Thesis Supervisor

Accepted by:_______

Ain A. Sonin
Chairman, Department Committee on Graduate Students

WEB Based Cost Estimation Models
for the Manufacturing of Advanced Composites
by

Joshua W. Pas

Submitted to the Department of Mechanical Engineering
on May 11, 2001 in Partial Fulfillment of the
Requirements for the Degree of Master of Science in
Mechanical Engineering

ABSTRACT

Presently there are many cost estimation methods and software available to the public for metal processing, but there are almost none (excluding proprietary) for advanced composites. As a result, our research group's objective has been to create Cost Estimation Models (CEMs) for the advanced composites. This thesis focuses on the modeling of the parts and processes as well as the implementation of the CEMs. With funding from the National Science Foundation (NSF), accessibility of the desired CEM deliverables to the public domain was paramount; therefore, the Internet technology has been used as our basic tool.

The CEM is a process-based cost estimation model using the first order dynamic law. The time constant, extensive velocity, equations and other parameters are based upon the work of [Neoh]. The goals of the CEM are to give the time and cost estimations as well as to assist the designer in evaluating the cost reduction strategies. Because the CEM also facilitates in production volume and process decisions, the CEM user can easily make comparisons in order to make the best implementation choices that will subsequently yield the relative optimal results. Fourteen shapes have been modeled with five processes (Automated Tow Placement, Forming, Hand Layup, Pultrusion, and Resin Transfer Molding). The CEMs are developed for both novice and expert users by providing default values that can be modified. The expert user can easily overwrite the default values and modify the process plan.

All calculations for the models use JavaScript which has been embedded into the HTML interface. XML databases have been implemented into the CEMs to store the parameters from previous work for the process, materials, and resources. In future updates of the models, the programmer will only need to update the database without needing to make any changes in the JavaScript coding. A number of beneficial tools have been included in the models to assist the CEM user along every step. Finally, the CEMs have been tested with very satisfactory results. The CEMs are published to http://web.mit.edu/lmp/www/composites/costmodel/.

Thesis Supervisor: Timothy Gutowski
Title: Professor of Mechanical Engineering

Acknowledgments

I would like to thank my research advisor, Professor Timothy Gutowski, for this appreciated support and guidance throughout the research. Special thanks are also given to my research partner, Sascha Haffner, for his many insights and useful suggestions.

Many people have made my two years at MIT memorable: Karuna Mohindra for frequent friendly help; Sam Truslow and Guido Beresheim for all the fun time spent together in the lab; and my many TSMIT friends for willingly taking the time off from study and research to play sports together.

Most importantly, I want to thank my parents, Kenneth and Jutharat, and sister, Yotkhwan, for their support, encouragement and caring. Their love has fed me again and again, making everything possible.

Table of Contents

Table of Figures

Chapter 1 Introduction

This chapter consists of six sections, which are:

1.1 Advanced composite materials

1.2 Composites processes

1.3 First order dynamic

1.4 Computerized modeling

1.5 Cost estimation model

1.6 JavaScript, HTML, and XML

1.1 Advanced Composite Materials

"Advanced composite materials are the ultimate "designer" material. By using various configurations of highly packed long slender fibers, one can obtain an enormous range of material properties. Furthermore, by changing the fibers and/or the matrix, intermingling the components, and designing the interface, the material options can become even more expansive, almost uncountable." [Gutowski]

Composite materials can be defined as the macroscopic combination of two or more distinct materials, having a recognizable interface between them. Composite materials contain a reinforcement (such as fibers or particles) supported by a binder (matrix) material. Composite materials were developed because no single, homogeneous structural material could be found that had all of the desired attributes for a given application. For example, Aluminum alloys provide high strength and good stiffness at low weight, good performance and have been the main materials used in aircraft structures over the years. However, both corrosion and fatigue in aluminum alloys have produced problems that have been very costly to remedy. The fiber-reinforced composites, which were developed initially to eliminate corrosion and crack formation, provide other substantial benefits to designers and manufacturers [Reinhart]. The term advanced is used to differentiate those with high-performance characteristics, generally strength and stiffness, from the simpler forms.

The behavior and properties of composites are determined by 1) the materials of which the constituents are composed, 2) the form and structural arrangement of the constituents, and 3) the interaction between the constituents. Of all the composite materials, the fiber type (specifically the inclusion of fibers in a matrix) has evoked the most interesting among engineers concerned with structural applications.

The majors factors contributing to performance are the orientation, length, shape, and composition of the fibers; the mechanical properties of the matrix; and the integrity of the bond between fibers and matrix. Of the above, the orientation of the fibers is the most important. The orientation governs the mechanical strength of the composite and the direction in which the strength is the greatest. The orientation of fibers in the matrix can be accomplished with either continuous or short fibers. Practically all fibers, both continuous and short, presently being used have a circular cross section. However, hexagonal, rectangular, polygonal, annular, and irregular cross sections appear to promise improved mechanical properties.

The other major constituent in fiber composites, the matrix, serves two very important functions: 1) it holds the fibrous phase in place, and 2) it deforms and distributes the stress to the high-modulus fibrous constituent under an applied force. The choice of a matrix for structural fiber composite is limited by the requirement that it has a greater elongation at break than the fiber. Since many reinforcements tend to be brittle, the matrix protects their surface against abrasion or environmental corrosion, both of which can initiate fracture. In order to accomplish a transfer of loads, and also reduce the chance of failure in the matrix, adhesion to fibers or other reinforcements must be coupled with sufficient matrix shear strength to sustain the loads. Coupling is typically produced by wetting the reinforcement with the matrix which is in a molten or low-viscosity state. Fiber composites are able to withstand higher stresses than either of their individual constituents because the fibers and matrix interact and redistribute the stresses. The ability to exchange stresses depends critically on the effectiveness of the coupling or bonding between them. [Schwartz]

Among the benefits of high-performance fibrous composites are these [Noton]:

- Ability to meet diverse design requirements
- Significant weight saving, ranging from 25 to 50% of the weight of conventional metallic designs
- Satisfying the high torsional stiffness requirements

- Outstanding corrosion resistance
- Numerous fatigue and fracture attributes
- Excellent impact and damage tolerance
- Improved dent resistance
- Like metals, the indefinite shelf life of thermoplastics
- Achievement of low thermal expansion, though it varies significantly with the matrix material selected
- Simplification of manufacturing and assembly because of part integration (joint/fastener reduction) which can reduce engineering, purchasing, and follow-up costs
- As compared to metals, cheaper cost of composites tooling, frequently two to five times cheaper, which significantly reduces amortization costs
- Lower freight costs

1.2 Composites processes

Automated Tow Placement (ATP)

ATP usually consists of a CNC multi-axis machine which deposits prepreg tape or tows onto the contour of a layup or curing tool. An early version of the ATP process layed up 3-inch tapes onto a flat surface. Current developments lay up rows of thin 1/8-in. tows over complex surfaces with the capability to follow nongeodesic paths. The new ATP machines, Figure 1.1, offer impressive, good repeatability, and good accuracy at a considerable up-front cost. The tape is stored on spools, which are carried along with the tape-laying head, as shown in Figure 1.2. The tape can be placed and cut in all possible directions with the limitations that the laying up of small concave radii is smaller than the compression roller on the layup head. Because starting up and slowing down is cumbersome, these machines are best for producing large shell structures, such as fuselage and wing skins. Typical layup rates are up to 10 lb/hr.

The major advantage of the ATP machine is that the fibers are not tensioned, enabling the nongeodesic paths. ATP is mainly used in the aerospace industry for the production of large, simple-to-moderately-complex parts, but it is not economical to layup small, complex-shaped parts. [Gutowski], [Haffner, 1]

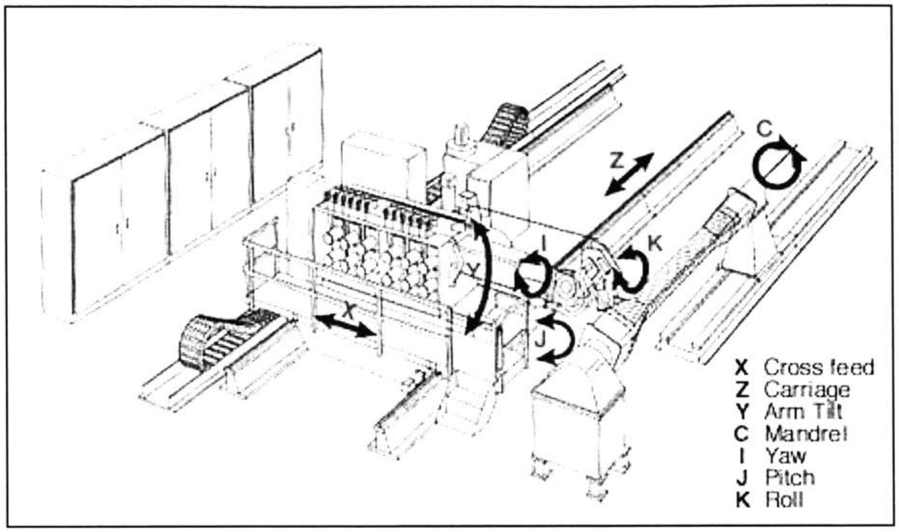


Figure 1.1 Tow Placement Machine [Cincinnati Machine]

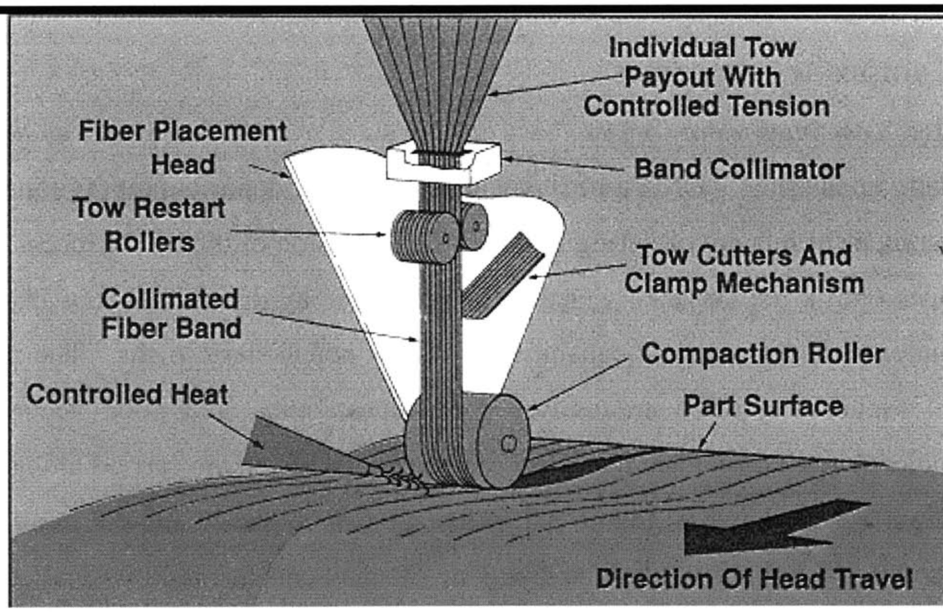


Figure 1.2 Fiber Placement Head

Forming

The Forming process is characterized by starting with the entire laminate in a flat state, then forming it into the final shape, offering a significant reduction in the production time from the Hand Layup process. The hydrostatic pressure shapes the stack of ply over a tool. Elastic diaphragms clamp the relatively stiff fibers with vacuum pressure, thus minimizing buckling and wrinkling in the formed part. Several derivatives of the process are single diaphragm forming, double diaphragm forming, and matched die forming. Once the part is formed, it is usually transferred to an autoclave or oven for curing.

Although more complicated, the diaphragm forming of composites is similar to the thermoforming of polymers. The Forming process can be integrated with a simple automated flat lamination center and provides a highly automated production system. Of the new composites processes, Forming has the lowest combined tooling and equipment cost. However, shape and size partly limit the Forming process. Figure 1.3 shows the schematic of the double diaphragm forming process. [Gutowski], [Haffner, 1]

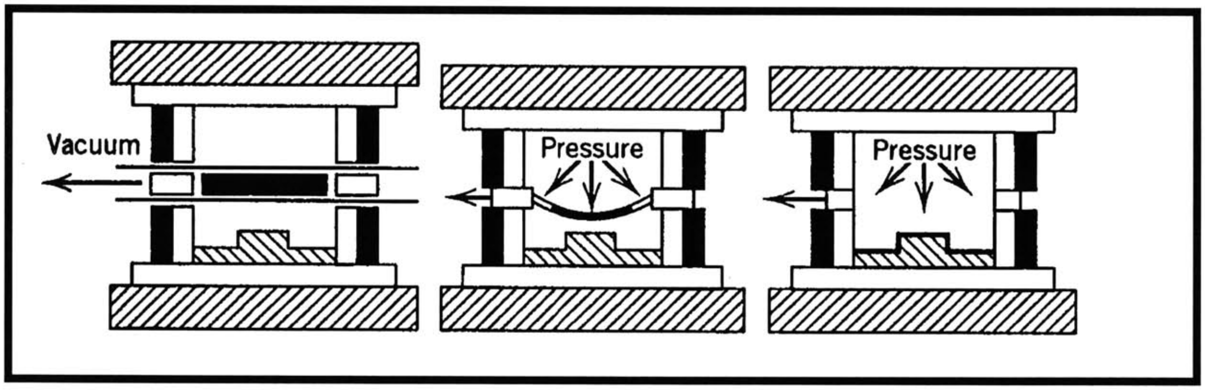


Figure 1.3 Schematic of the Double Diaphragm Forming [Gutowski]

Hand Layup

By far, the most important process in terms of current aerospace production, is Hand Layup of prepregs and autoclave cure. In spite of steady progress of replacing Hand Layup with automated processes, it still persists as the method by which at least half of all advanced composite aerospace structure is made because it is extremely flexible and thus capable of making a wide variety of shapes.

Composite fibers are manually deposited layer by layer onto a tool which gives the part its shape. The operator cuts each ply, removes possible release films and places the new ply in its predefined location. Therefore the correct orientation of the fibers and absence of wrinkles could be ensured. For high performance parts, the accuracy of ply deposition can be enhanced by lay up aids such as pins, marks, and laser projectors.

Figure 1.4 shows the schematic of a vacuum bag Hand Layup which consists of:

- Vacuum bag which removes trapped air and volatiles. It assists in the resin flow and densification of a composite part before and during curing. It is also used during the debulking step.
- Bleeder plies which absorb excess resin during curing. They are crucial to achieve correct fiber-to-resin ratio.
- Barrier film which is a non-adhering layer that prevents excess resin flow from clogging breather plies and vacuum lines.
- Breathers which allow uniform application of vacuum and removal of air and volatiles.
- Release films which are placed next to the tool surface to pass volatiles and trapped air and to facilitate the release of the part from the tool.
- Peel plies which protect composite parts from damage during the layup step. They also absorb volatiles from the laminate during curing.

The Hand Layup process does not entail the large capital investment usually required by the automated processes. Nevertheless, as production volumes increase and economic pressures intensify, Hand Layup will gradually be replaced or modified by the use of new automated technology. Because the process is both tedious and manual, the results can be variable and subject to operator differences. As a result, a thorough inspection is required on all structural parts.

There are no limits for Hand Layup in terms of part shape. Part size is only limited by accessibility and sometimes material out-time. The manual lay up rate generally ranges from 0.002 to 2 lb/hr. Hand Layup is commonly used for low volume or prototype production of 1 to 30 parts/month depending on size. Aerospace and boat building are typical applications of the Hand Layup process. [Gutowski], [Haffner, 1]

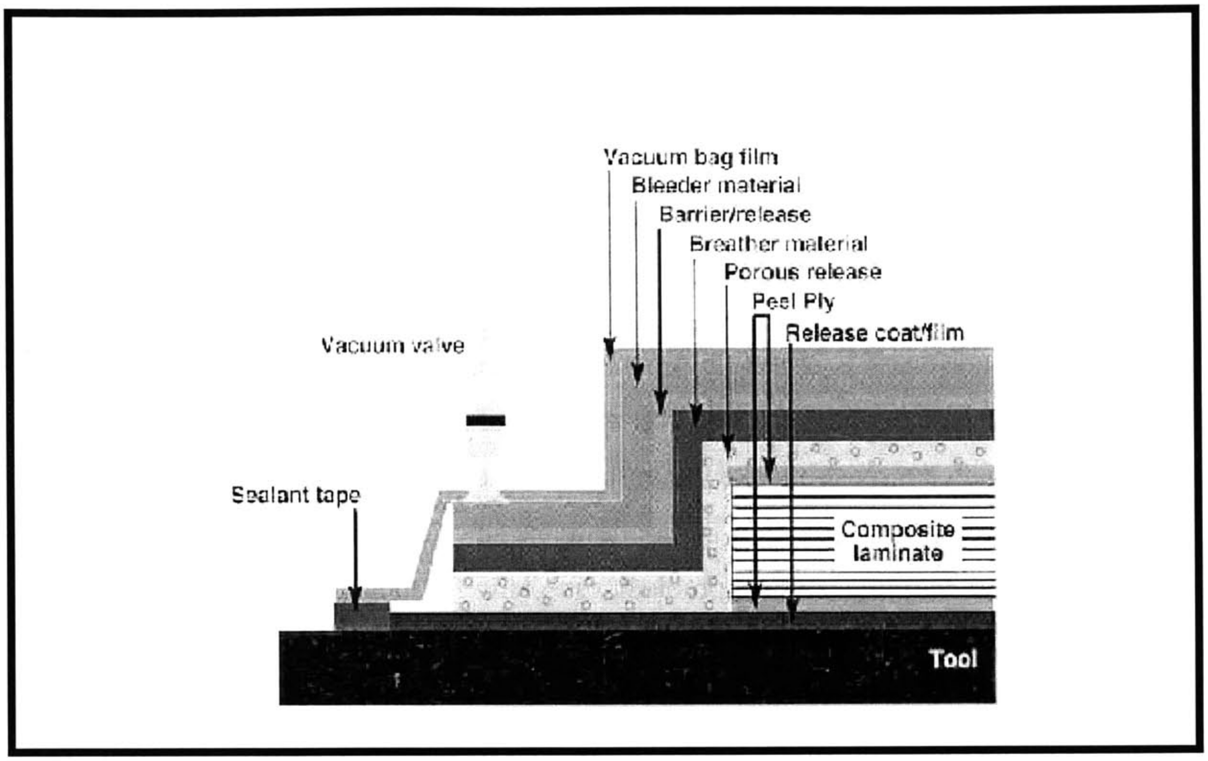


Figure 1.4 Schematic of the Vacuum Bag Hand Layup [Gutowski]

Pultrusion

Pultrusion in its basic form involves the pulling of fiber through a wetting station and then into a heated die, where the resin is cured. The part is then sawed to length, on the fly. The process steps include material delivery, wetting, material pre-shaping, curing, pulling, and cut-off activities as shown in Figure 1.5. Resin dip bath is most commonly used for the resin impregnation. The wetting speed depends upon the pretreatment of the fibers and on the resin formulation. The Pultrusion dies are categorized into two types which are forming die and curing die. The forming die is located immediately after the impregnation process whereas the curing die is attached to the forming die. Die heating is the most critical control parameter in the Pultrusion process. The thick section requires the longest heat input which slows down the Pultrusion rate, thus the design of wall thickness is very crucial. Additionally, it is best to design uniform cross-sectional thickness with 2-3% allowed cross sectional shrinkage and reduce warpage by symmetrical design. During the start up and shutting down periods, it is necessary to provide a cooling method at the front of the die to prevent early gelation of the resin system. Radio frequency heating and traditional die heating combined can significantly increase the running speed. [Gutowski]

The advantages of this process are continuous production, low labor requirements, low material scrap rate, and the eliminate supported materials (breathers, bleeder cloth, separation film, bagging film, edge tape). The speed ranges between 1 in/min to 12 in/min.

"The Pultrusion process is relatively productive and is among the lowest-cost advanced composites processes. There are, however, some limitations on the fiber orientations that can be used, and care must be taken to ensure that fibers do not move around in the die during the process. The long dies are not desirable since they increase the pulling force. Straightness and warping can be a problem, especially for long, thin parts. One solution could be the postcure on a full-length tool." [Gutowski]

The Pultrusion process is typically used for the mass production and is increasingly used as structural elements of larger constructions such as buildings, bridges and electrical towers in particular because of the insulation capabilities of pultruded glass fiber. [Haffner, 1]

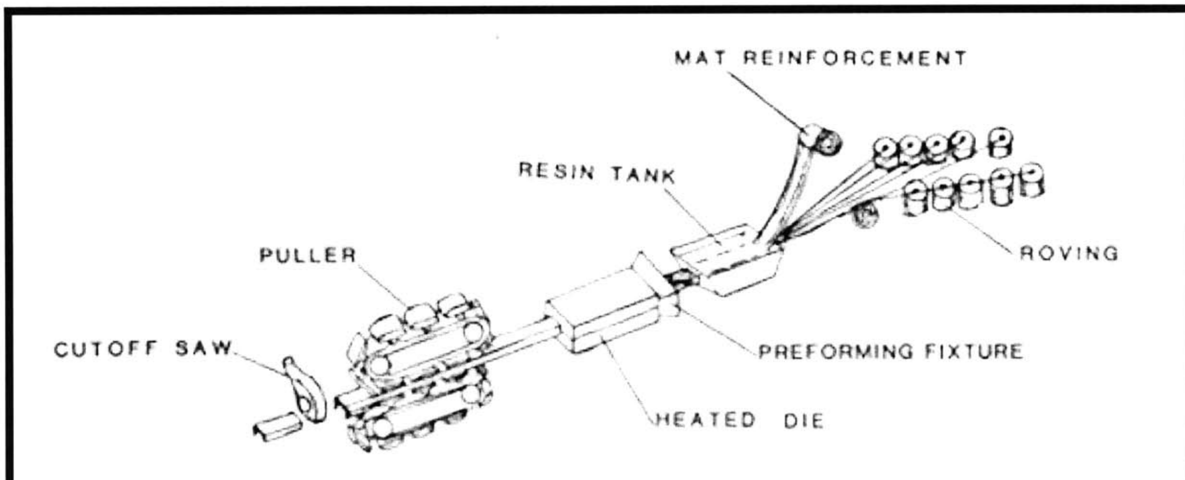


Figure 1.5 The Pultrusion Process

Resin Transfer Molding (RTM)

RTM refers to a group of processes that inject resin into a fiber preform captured in a closed tool. The preform is often preshaped and produced by secondary processes such as hand layup, braiding or weaving. The resin reaction may be initiated by heating (usually from the mold wall) or by mixing in a reacting compound prior to injection (SRIM). After tightly closing the mold, liquid resin is injected under pressure (approximately 100 psi) to impregnate the fiber reinforcements. The part is often cured in the mold; giving smooth surface finishes on both sides and good tolerances.

The application of the process for high performance products can be limited by the slightly lower fiber volume fraction inherent to the process. In terms of size there are certain limits because of insufficient impregnation. However, special derivatives (SRIM) of the process are also suitable for large structures. The RTM process, along with stamped performs, is of interest to the automotive industry because of its potentially high production rate. The tooling costs are usually high, but the closed tool protects workers from any potential exposure to the curing resin. One attractive feature of RTM is its ability to make very complex parts which can reduce or eliminate the need for assembly. RTM appears to be best suited for medium volume, small to medium sized complex parts. Production runs of 100 to 1000 parts/month are considered economical. Figure 1.6 shows the RTM process. [Gutowski] and [Haffner, 1]

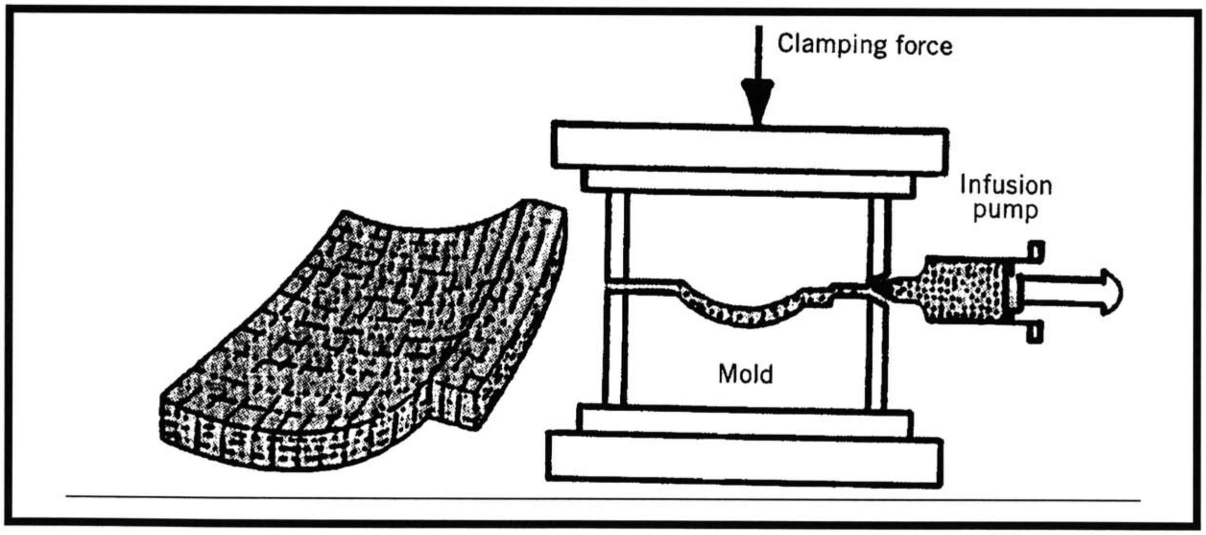


Figure 1.6 The RTM Process [Gutowski]

1.3 First order dynamic

Based upon a reviewing considerable data for both composites fabrication and for machining operations, [Neoh] observed that manufacturing operations (both humans and machines) can be represented as dynamic systems with first order velocity response to a step input and thus are amenable to physical modeling, as shown in Equation 2.1 and plotted in Figure 1.7.

$$V = Vo(1 - e^{-td/\tau}) \qquad \text{Equation 2.1}$$

This approach has the advantage of characterizing the process using the physical quantities "Vo", the steady-state velocity, and "τ" the dynamic system time constant. Both are dimensionally correct, and have meaningful physical interpretations.

The use of dynamic models requires accurate physical description of the system. Moreover, the usefulness of such models depends on the appropriateness of the modeling assumptions. Examples of dynamic models can be found in [Neoh].

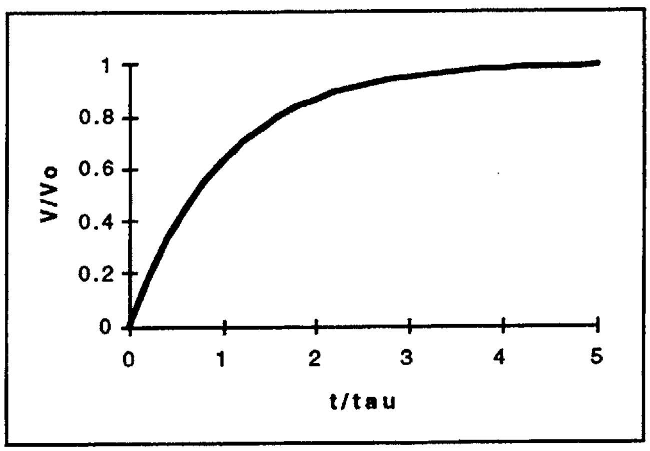


Figure 1.7 Dynamic Systems with First Order Velocity Response to Step Input

"Integration and mathematical rearrangements of the first order dynamic model leads to the hyperbolic model, which gives the desired size time dependence. Equation 2.2 explains the hyperbolic model. For extensive processes, where "x" is the extensive variable (length, area, or volume) the manufacturing time could be successfully described by only two factors that represent the process; the process rate Vo and a time constant τ. As for the effects of the complexity, the hypothesis of linear relationship between the information content of a part, "I" and the first order parameters Vo and τ was supported. Equation 2.3 explains the effects of complexity." [Haffner, 2]

$$(\text{time}/\ \tau) = [(x/(Vo.\ \tau)+1)^2 - 1]^{0.5} \qquad \text{Equation 2.2}$$

$$\tau = \tau o + b.I \quad ; \qquad (1/V) = (1/Vo) + (I/c) \qquad \text{Equation 2.3}$$

1.4 Computerized modeling

To have a successful computer model, the designer should meet the goals of giving immediate feedback, reducing the number of steps required to accomplish a task, providing full function with small number of objects, increasing user control over the product, reducing the potential for exceptions, reducing the effect of exceptions, and providing interaction technique suited to the needs of novice and expert users.

In the past several years, as consumer products have become more computerized and complicated, a greater emphasis has been placed on the user-friendliness or the products. The programmer must understand the users and then develops a conceptual model to design the task. This conceptual model is conveyed to the user through the display representation so that the user must form a mental model of the task that is as close to the conceptual model as possible. The interface design is the tool uses to link the conceptual model to mental model. [Norman]

"The WEB is the ultimate customer-empowering environment. He or she who clicks the mouse gets to decide everything. As a result of the ease of going elsewhere, web users exhibit a remarkable impatience and insistence on instant gratification. If they cannot figure out how to use a website in a minute or so, they conclude that it won't be worth their time. And they leave." [Nielsen]

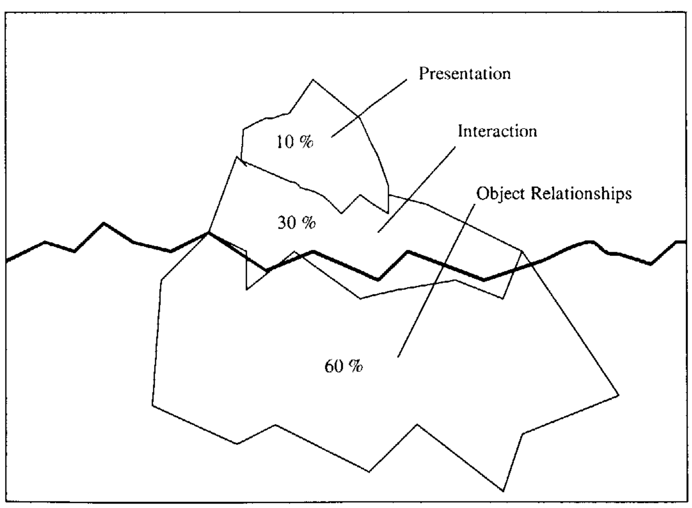


Figure 1.8 The Designer's Iceburg

The most important factor in making a good computerized model are the presentations and interactions as can be explained by the designer's iceberg, show in Figure 1.8. The object relationships, which include relationships, properties, and behaviors, are the major part of the model (60%). However, the user is not aware of this. S/he is only aware of the visual representations which is the smaller part of the model (combined to 40%). Therefore, it is very important not only to make a well-functioning model but also to create a good interface design. The goals of the user interface design are to increase the user's productivity, to increase user's satisfaction, and to reduce error rate.

In creating computerized models, an iterative development process is always followed. The process, Figure 1.9, includes research/planning, designing, prototyping, and testing. The activities at the research and planning phrase are gathering requirements, learning about users and their tasks, and matching requirements to the tasks. Next the programming activities occur at the designing and prototyping phrases. Finally, testing is performed to ensure the model's workability.

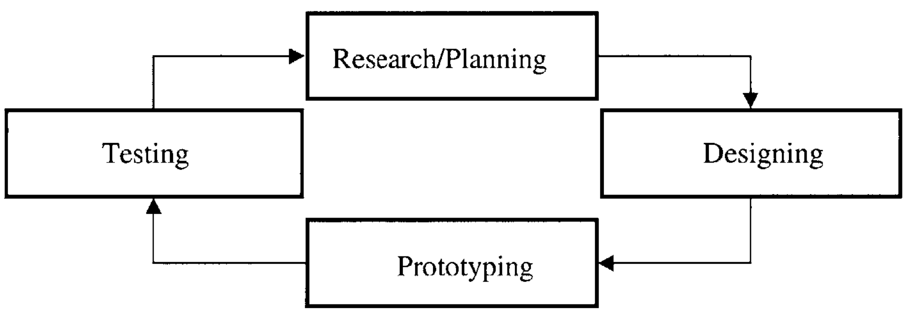


Figure 1.9 Iterative Development Process

1.5 Product cost estimation

A logical criterion for evaluating the design would certainly be the cost required to bring each design into being and the manufacture of the product. Cost is also used to determine the most economical operation or sequence of operations for manufacturing a product and can be used as a means for establishing a cost-reduction program aimed at manufacturing the product so that it can be priced more competitively. Cost can be classified in different ways based on their relationships to the production volume and the nature of the manufacturing operations. The most logical way is to split them into two groups: capital costs and operating costs. Capital costs are incurred because of buildings, production machinery, and land. In cost estimation, the buildings and machinery are depreciable, whereas land is not. Operating costs are "running" costs that reoccur when the plant is in operation. Another way to classify costs is to view them as belonging to one of two categories: fixed costs which are independent of the production volume, and variable costs which are dependent on it. [El Wakil]

"The direct manufacturing cost is given by: $C = C_E + C_L + C_M + C_T$, where C_E is the equipment depreciation, maintenance and operating cost, C_L is the labor cost, C_M is the material cost, and C_T is the tooling cost." [Krolewski] Labor can be either direct or indirect. Direct labor is explicitly related to the process of building the design, whereas indirect labor involves the work of foremen, stock-room keepers, and so on. Labor rates are dependent on the skills of laborers and the type of tasks. Skilled laborers, such as equipment programmers, generally command a higher wage than other workers. As for indirect manufacturing costs, several important ones are work-in-progress inventory, floor space, indirect labor for quality control, supervision, shop scheduling and engineering, and rework and scrap which are relevant to the economic comparisons. Indirect labor costs are generally pooled and distributed to a part on the basis of direct labor hours, using the indirect multipliers. [Krolewski]

As for advanced composites, the manufacturing cost can be broken down into four major subcomponents: material costs, processing costs, assembly costs, and inspection costs. Material costs tend to be expensive and are usually driven by the fiber cost. Processing costs are driven by the part design, the production volume, and how the process is run. Design issues include the size and complexity of the part. The process operation includes such issues as operator skill and position on the learning curve. The three major categories of assembly methods are 1) mechanical assembly using rivets and bolts, 2) cobonding of precured parts (similar to adhesive bonding), and 3) cocuring (the integrated cure of an entire assembly or the integrated cure of some procured parts with some uncured parts). The assembly cost varies significantly between these three methods. As for inspection, high performance parts often require 100% inspection. [Gutowski]

1.6 JavaScript, HTML, and XML

According to [Mathieu], the Internet applications in manufacturing have the potential to transform and improve significantly all stages of manufacturing operations--from technology and market assessments to design for manufacturability, R&D, and after sales support. The Internet has improved the competitiveness of many manufacturing organizations by making available best manufacturing application tools, knowledge bases, product information, and training materials. It minimizes the risk of a manufacturing organization remaining isolated and incapable of integrating applications, and interacting with other companies, suppliers, and customers in a timely and cost effective manner. In manufacturing, both efficient and effective management as well as the manipulation and use of information are essential to economic vitality and growth. The most effective web sites are the ones with access to corporation databases. To improve as an industry, each manufacturer should learn from the others by means of communication. For the manufacturing organization, five basic Internet strategies are suggested:

Strategy 1 Communication with customers/distributors

Strategy 2 Communication with suppliers/vendors

Strategy 3 Collaborating with other organizations

Strategy 4 Communication within the organization

Strategy 5 Learning from outsiders (CEMs included in this strategy)

The Internet is a very important tool for the use of Virtual manufacturing (VM) which is an integrated environment that enhances all levels of decision-making and the control in product and process design, process planning, production planning, and shop floor control. Three major types of virtual manufacturing can be identified, namely design-centered VM, production-centered VM, and control-centered VM. Our CEMs are categorized in the production-centered group because they provide an environment for process plan generation, production plan generation, resource requirement planning, and plan evaluation.

There are numerous computer languages for the Internet. JavaScript, HTML, and XML have been selected for our CEMs because of their suitability to the objectives and applications of the CEMs.

JavaScript

One of the motivations behind JavaScript was the recognition for logic to exist on the client, not simply on the server. With all logic on the server side, all processing is forced to go to the server, even for simple tasks such as data validation. Providing logic within the browser empowers the client and makes the relationship a true client/server arrangement.

Java is a step in this direction, but it's implemented as an adjunct to HTML itself and is not intended to be integrated from a language standpoint. A high level, client-side scripting language seemed like a natural missing piece in the Web development tool arena. Accordingly, the JavaScript language was developed to solve this problem.

Perhaps the most important JavaScript fact is its marriage with HTML. There is hardly any separation between the two. JavaScript code is usually housed within HTML documents and executed within them. Additionally, by itself, JavaScript has no user interface. Instead, it relies on HTML to provide its means of interaction with the user. JavaScript uses HTML as a means of jumping into the Web application framework. It also extends the normal capabilities of HTML by providing events to HTML tags and allowing event-driven code to be executed within it. As with most scripting languages, JavaScript is interpreted at runtime by the browser before it is executed. It is not compiled into an actual program but remains part of the HTML document to which it is attached.

JavaScript is an object-based language and event driven. The programmer has to work with objects that encapsulate data (properties) and behavior (methods). Much of the JavaScript code will be in response to events generated by the user or the system. The JavaScript language itself is equipped to handle events. HTML objects, such as buttons or text fields, are enhanced to support event handlers. JavaScript is multifaceted and can be used in a variety of contexts to provide a solution to a Web based problem. Some of the primary purposes are to enhance and enliven static HTML pages, validate data without passing everything to the server, serve as a building block for client/server web applications, develop client-side applications, and provide database connectivity. [Wagner]

Hypertext Markup Language (HTML)

HTML is the language used to prepare hypertext documents, which are the ones distributed on the WEB and which the user actually sees. HTML contains commands, called tags, to mark text as headings, paragraphs, lists, quotations, emphasizes, and so on. It also has tags for including images, fill-in forms to accept user input, and links to other resources. The links allow user to click on a string of highlighted text and access new document, images, or movie file from around the world. The URL, which specifies where the document is, is a flexible scheme and, in combination with HTML, yields an incredibly powerful package for preparing a web of hypertext documents linked to each other and to Internet resources around the world.

HTML is designed to specify the logical organization of a text document, with important extensions for hypertext links and user interaction. It requires that the WEB-designer construct documents with sections of text marked as logical units, such as titles, paragraphs, or lists, and leave the interpretation of these marked elements up to the browser displaying the documents. This model builds enormous flexibility into the system and allows browsers of different abilities to view the same HTML documents. [Graham]

Extensible Markup Language (XML)

XML, derived from the SGML language, has the ability to work with HTML for data display and presentation for the WEB-delivered data. Not only does XML support the basic hyper-linking found in HTML, but it also takes the concept further with extended linking. XML includes a specification for a style language called Extensible Stylesheet Language (XSL) which allows the programmer to create a template of various styles or combinations of styles and apply them to elements in a document.

If HTML is about displaying information, XML is about describing information. XML is a standard language used to structure and describe data that can be understood by different applications. The power of XML is its ability to separate the user interface from the data. XML is self-describing, which means the document contains the set of rules to which its data must conform. It can be used as a data interchange format, web data, and data storage for information that might get used in many different ways. One can imagine having an HTML page in which none of the content is located on the page itself. Instead, the content is stored in an XML file, and the HTML page is used simply for formatting and displaying. The formatting, layout, and so on are dependent upon the applications that use the data and are not attached to the content itself. Furthermore, the application code that displays the data needs to be written only once, and can then be used to display any number of articles.

People usually think of XML as a replacement for HTML. Although this might be partly true, the two languages are more likely to be complementary to one another than in competition. For cases in which XML is used to structure and describe data on the web, HTML will likely be used to format that data. Since all the content can be kept separate from the HTML code, the content is easy to change and work with. HTML document can be used as a template to create many documents containing different data. Although XML replaced portions of the HTML code because the HTML code no longer stored any of the data for the document, the XML code did not constitute a total replacement of the HTML code. The creator of XML defined 10 goals for the language including the following: the XML should be straightforwardly usable over the Internet, it should support a wide variety of applications, it should be easy to write programs which process XML documents, XML should be human-legible and reasonably clear, and XML documents should be easy to create. [Pardi]

Chapter 2 Cost Estimation Model (CEM) Structure

This chapter is broken down into four sections which are:

2.1 Objective

2.2 Shapes and Processes

2.3 General information on the CEM

2.4 Equations

This chapter aims to explain the structure of the cost estimation model (CEM) as a big picture. It is expected that the user can understand what the CEM is and how it works after s/he completes this chapter. Chapter 3 discusses the example of the Hand Layup of a simple curved part in detailed fashion. Chapter 4 will then discuss the implementation of the CEM, including the coding. Finally, Chapter 5 will discuss the process plan, the coding, and the issues concerning the other four processes: ATP, Forming, Pultrusion, and RTM.

2.1 Objective

The main objective of the CEM is to guide the designer in producing advanced composites products. At the moment, advanced composites manufacturing is not as well-known a field as is the case for metal manufacturing, largely because there are no time or cost estimation equations or software available to the advanced composites industry whereas there are many for metal processing. Companies frequently have cost models but they are proprietary and not available to general public. Although companies do understand their processes, they frequently are not acquainted with other alternatives in the industry. With the CEM, the designer is assisted in the evaluation of cost reduction strategies. S/he can gain the confidence to make decision early on during the design phase as a result of the information obtained from the models.

The CEM is a process based cost estimation using the first order dynamic law. The time constant, extensive velocity, equations, and other parameters are based on [Neoh]. Additionally, the physical models are constructed to obtain the absent parameters. With funding from the National Science Foundation (NSF), the desired deliverables for the CEMs must be accessibility to the public domain. In making the CEMs available to the advanced composites industry, the research group decided to construct models using the Internet technology as the basic tool. All calculations for the models used JavaScript which was embedded into the HTML interface. JavaScript processes the information at the user's computer in contrast to many other languages that perform the necessary executions at the server. XML databases are also implemented into the CEMs to store the parameters for the process, materials, and resources. The main benefit of the XML database is that in updating the models, the programmer only needs to update the database without having to change the JavaScript coding. The XML database and HTML supplement one another.

The benefits to the CEM user are the following:

- Obtain the realistic cost for a particular part and process, with easy-to-use models
- Assist the designer in selecting the process
- Assist the designer in understanding the steps involved in the particular process
- Give the designer understanding about the limitations to each process
- Help the designer decide on the production volume to be used and advise the optimal process choice for the desired production volume
- Obtain the time and cost involved in each process step for the manufacturing systems planning
- Give the designer reference to our databases
- Enable the designer to evaluate the sensitivity to the results of CEMs regarding changes in the parameters since the results are based on many assumptions about part dimension parameters, material parameters, and labor parameters

The future work for our research group is to update the material information, add tooling, equipment, and assembly estimations to the current CEM, and create manufacturing systems planning with the assistance of the CEM. Additionally, the effect of learning curve to CEM is to be studied because humans learn and get better at the task and that there are economies of scales for large production. A successfully tested tooling addition was easily embedded into the CEM structure, giving the researchers the confidence that the others could be successfully implemented.

2.2 Shapes and Processes

After researching for typical shapes in composite parts, fourteen shapes have been selected. They are then modeled with five processes (ATP, Forming, Hand Layup, Pultrusion, and RTM) for a total of seventy models. However, not any particular process can make every shape. For example, the curved part cannot be made by the Pultrusion process since Pultrusion is done longitudinally. Moreover, the straight I profile, the straight T profile, the curved I profile, and the curved T profile cannot be processed by ATP since the roller cannot be placed in that manner.

We came up with the system of classifying the manufacture-ability to common (C), possible (P), and not common (NC). The definition of (C) is that it is technically possible and often seen in market place. (P) means technically possible and (NC) means not observed in market place and/or present technical issues. With this ranking, only the fifty-six models that are either C or P are to be completed.

Figure 2.1 shows the selection matrix and Figure 2.2 shows the shapes to be processed.

Shape\Process	Hand Layup	Forming	RTM	ATP	Pultrusion
Shape 1	C	P	P	P	P
Shape 2	C	P	P	P	P
Shape 4	C	P	C	P	NC
Shape 5	C	C	C	P	C
Shape 6	C	C	C	P	C
Shape 7	C	P	C	NC	C
Shape 8	C	P	C	NC	C
Shape 9	C	P	C	P	NC
Shape 10	C	P	C	P	NC
Shape 11	C	NC	C	NC	NC
Shape 12	C	NC	C	NC	NC
Shape 13	C	NC	P	C	P
Shape 15	C	NC	P	P	P
Shape 16	C	P	C	C	NC

Figure 2.1 Selection Matrix (C=Common, P=Possible, NC=Not Common)

From Figure 2.1, it is apparent that Hand Layup can make almost any possible shape since the ply is deposited manually over the tool. The I-profile is done by cocuring two C-profiles together and T-profile is done by cocuring two L-profiles. Making the Shape

11 or 12 is not common for the Forming process since the part can be too curved. The shape 13 or 15 cannot be done straightforwardly with the Forming process, but can be made with two halves and then assemble together. The RTM can make almost every part, provided that the part is not too big for the mold and the oven. The ATP process has limitations on the rollers positioning and the tools for holding the part. Because the Pultrusion process pulls straight parts with a constant cross section, curve parts cannot be pultruded.

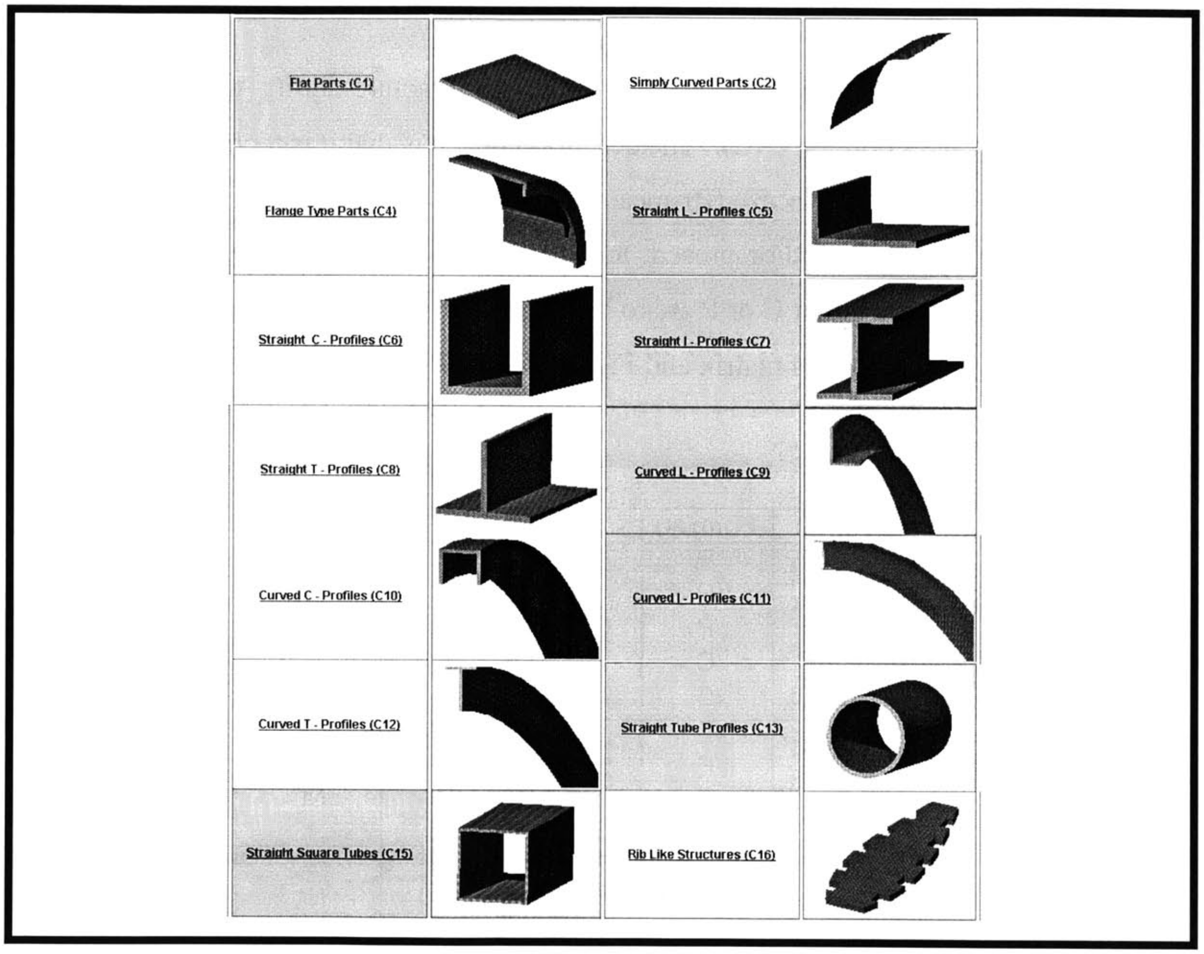


Figure 2.2 Shapes in CEM

Selecting the most appropriate manufacturing process in terms of technological feasibility and cost for a component design is one of the most important decision-making tasks. Failure to get it right normally results in components that are of variable quality and/or expensive to make. The greatest opportunity in manufacturing design occurs at the initial design stage. While there are possibilities when a product in production is to be modified, there are many additional constraints. See Figure2.3. The selection of the most appropriate process depends upon a large number of factors, primarily product volume, processing times, equipment costs, tooling costs, labor intensity and work patterns, energy consumption and other overhead costs, material costs and availability, tolerance requirements, and process waste. The CEM is designed to assist the user in making this selection by considering some of the above necessary factors.

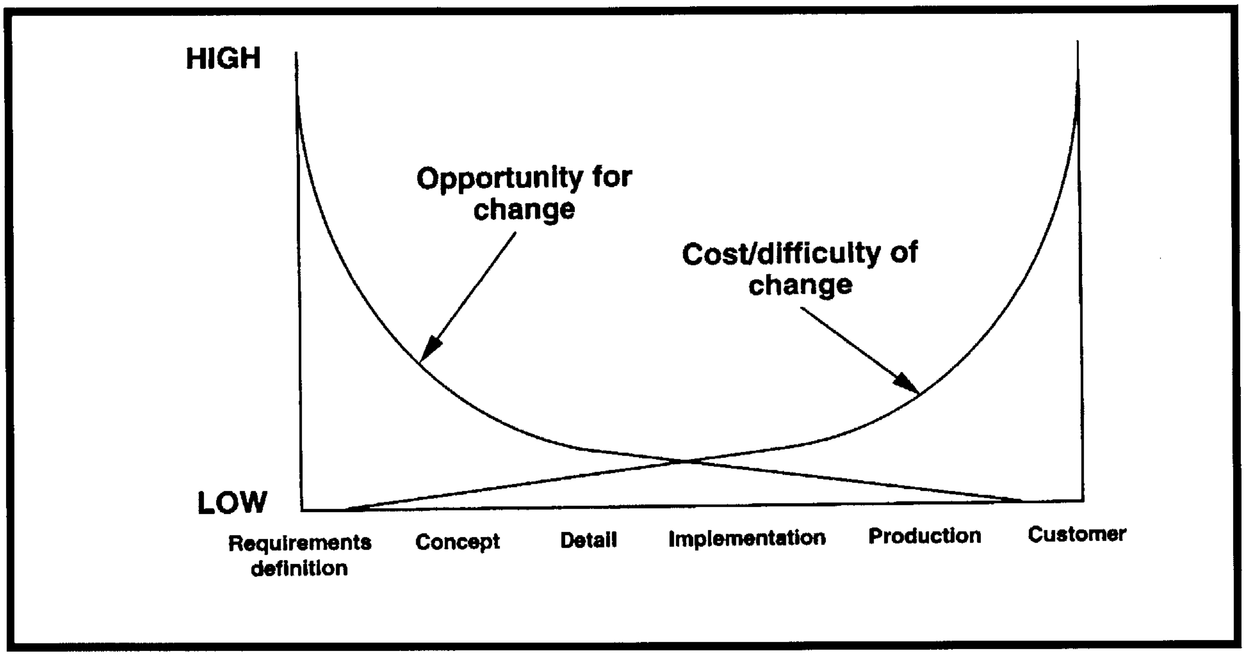


Figure 2.3 The Opportunity for Change [Swift]

With part descriptions available to the user, s/he can click at the part name to link it to the part details. These pages, written by Haffner, give information on the production volume, process, tooling, issues, and comments for the designers. The process descriptions, also written by Haffner, are available at the selection matrix. Just by clicking the link, the user can find out much information about the processes.

2.3 General information on the CEM

How to use the CEM

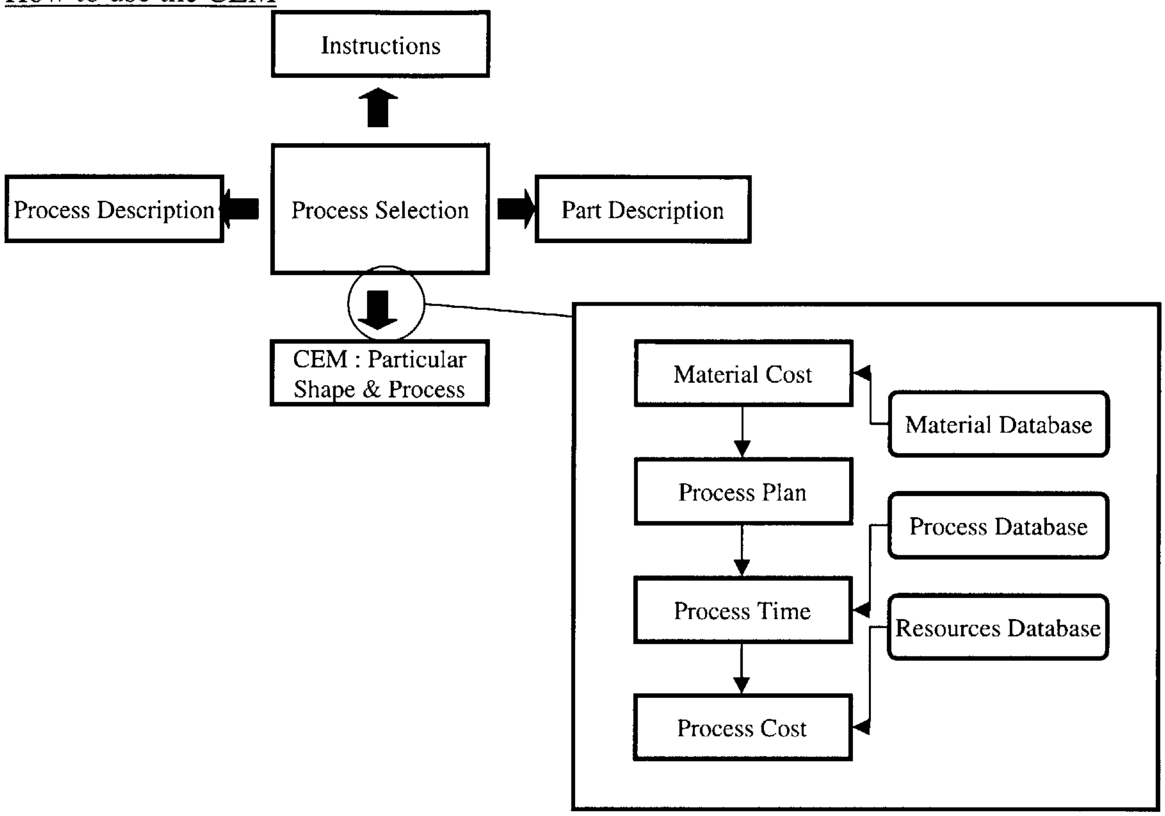


Figure 2.4 How to use CEM

Base on the Figure 2.4, the starting page is the process selection matrix page. From here, the user can be directed to four destinations which are the process description page, the part description page, the instruction page, and the CEM page. Many decisions can be made with this process selection matrix. For example, if the CEM user wants to produce some particular parts, without much understanding of the processes, s/he can narrow the choices down by looking at the classification (C, P, and NC). If the user wants to learn more about each process capable in making his/her part, s/he should click the "Process Description" link. If the user wants more information on the part itself, then clicking the "Part Description" link is the solution. Figure 2.5 shows the process selection matrix.

Click here to go to Instruction Page

Process / Part	Hand Layup	ATP	Forming	Pultrusion (Done!)	RTM (Done!)
Flat Parts (C1)	C (1.23)	P (2.12)	P	P (1.29)	P (2.02)
Simply Curved Parts (C2)	C (1.24)	P (2.12)	P (2.22)	P (1.30)	P (2.02)
Flange Type Parts (C4)	C (1.24)	P	PNC	NNC	C (2.01)
Straight L - Profiles (C5)	C (1.24)	P? (2.07)	C	C (1.30)	C (1.31)
Straight C - Profiles (C6)	C (1.25)	P?	C	C (1.30)	C (2.01)
Straight I - Profiles (C7)	C (1.26)	NC	P	C (1.30)	C (2.01)
Straight T - Profiles (C8)	C (1.25)	NC	P	C (1.30)	C (2.01)
Curved L - Profiles (C9)	C	P?	P (2.22)	NC	C (2.01)
Curved C - Profiles (C10)	C	NC	P?	NC	C (2.01)
Curved I - Profiles (C11)	C	NC	P?	NC	C (2.01)
Curved T - Profiles (C12)	C	P?	P?	NC	C (2.02)
Straight Tube Profiles (C13)	C	C (2.12)	NC	P (1.30)	P (2.02)
Straight Square Tubes (C15)	C	P?	NC	P (1.31)	P (2.02)
Rib Like Structures (C16)	C	C	P	NC	C

If you have comments or suggestions, please email me at joshpas@mit.edu
This information was last updated March 02, 2001

Figure 2.5 Process Selection Matrix

Instruction page

Figure 2.6 shows the Instruction page. The model is broken into the following sections: picking the process, picking the part, material selection, dimensioning, quantity section, material section, process time, time breakdown, and cost calculation. If the user needs assistance with any particular section, s/he can obtain step-by-step instructions simply by clicking the link to that section.

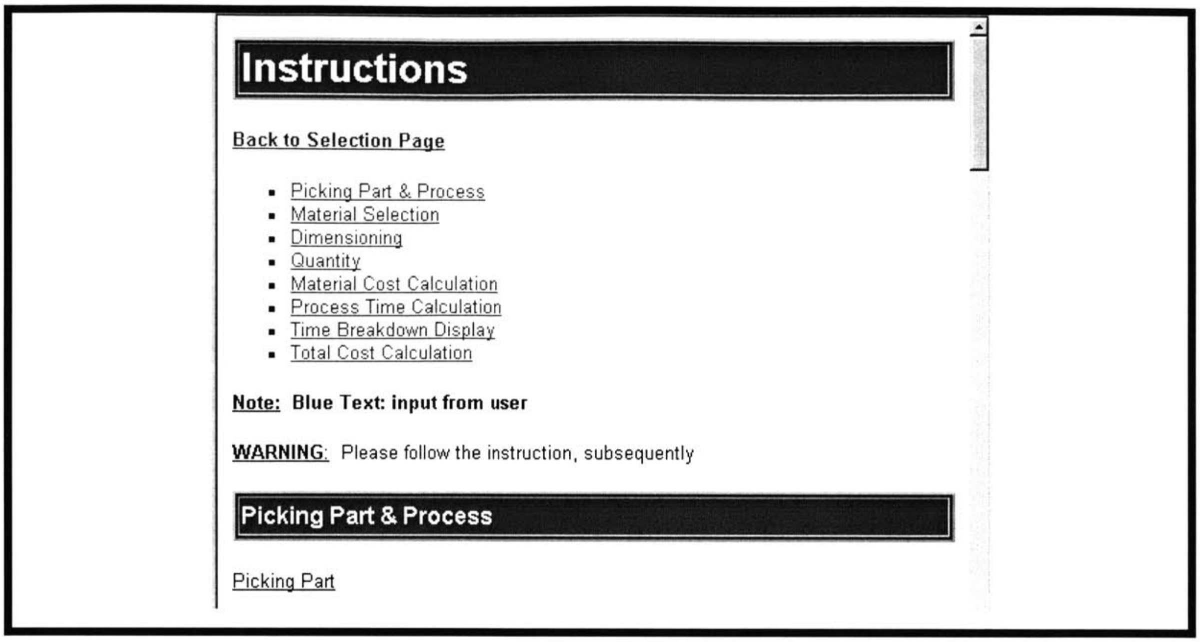


Figure 2.6 Instructions Page

Process pages

By linking to the matrix selection page, the user can benefit greatly from the process details. These pages, which were developed by [Haffner, 1], give a brief summary on the process descriptions, material, tooling, and applications.

Part pages

Again, these pages are linked to the matrix selection page. [Haffner, 3] provides details on the production volume (small, medium, and high), part size, processes, tooling, and issues for making a particular part.

Useful tools

Tools that increase the usability of the models have been constructed so that the model can be used conveniently without causing confusion. The tools in the CEM are the XSL database display, the individual process parameter HTML page, the time breakdown table, and the quantity classification.

1. Database display. The user can conveniently see all the information within the databases. XSL shows the parameters in table format for ease of usage. The user is able to scroll up or down to obtain any information contained in the database.
2. Individual process parameter HTML page. Since the user can select or deselect the process step, it is beneficial to see the parameters which are used in the process step. If additional steps are required, the user should be able to see the parameters used in the other process steps in order to benchmark the parameters. The user needs to input the process step ID and then click "Enter" to display all the parameters.
3. Time breakdown. Time, broken into non-recurring and recurring, is displayed for each main step. With the select and deselect feature, the user can acquire the time for each process step by deselecting every steps but that particular step.
4. Quantity classification. The total quantity is the product of the multiplication of part made per setup and number of setups. Even with the same total quantity, having different parts per setup or the number of setups can result in significant differences in process time and cost. The number of setups is responsible for the non-recurring time and the part made per setup is responsible for the recurring time which are delay and operation time. For the non-recurring time to get distributed to all the parts, it is best to have the smallest number of setups. Nevertheless, many times this cannot be done because of certain conditions, such as one setup being needed for every eight-hour shift. Different company has different ways of producing so how one allocates the production quantity is frequently different. An experiment was performed on the production quantity to differ the part per setup to make a total of 100 parts. The Hand Layup of flat panel part (width = 12 inches, length = 60 inches, and thickness = 0.125 inch) by the Fabric Std. Mod-3K-70-pw 42" untoughened was used in the tests. Figure 2.7 explains the relationship between the part per setup and the number of setups in making a total of 100 parts.

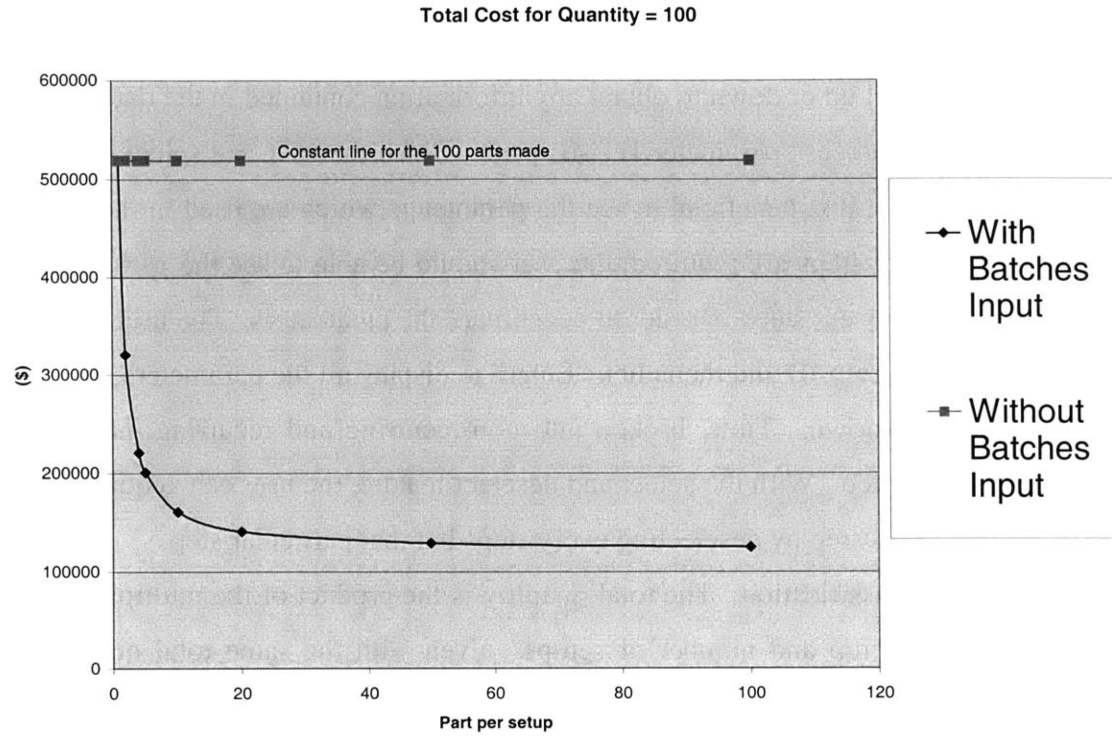


Figure 2.7 The Relationship of Quantity

As one can easily see, the two lines meet at the part per setup of 1 with 100 setups. As the part per setup goes up and the number of setups goes down, the cost decreases significantly. Producing parts with a part per setup of 50 and 2 setups differs to a part per setup of 1 and 100 setups by 75%. Thus, one can see that the classification of quantity to part per setup and number of setups is very meaningful.

Implementation of the CEM

Early in the project, the implementation of the CEM was carefully planned. The sequences of the topics are, respectively in this order: 1) HTML; 2) forms (for user interaction); 3) JavaScript; 4) XML databases; 5) useful tools; 6) error testing; 7) usability testing; 8) instructions; 9) final error testing; and 10) publishing. Topics 1), 2), 3), and 4) will be discussed in Chapter 4.

Error testing

Every experiment or implementation needs error testing to assure that all activities are executed correctly. The same was true for the CEM. Spreadsheets were developed specifically for this purpose. Each process had its own spreadsheet with the same equations with the CEM to check at every process step. The estimated parameters were entered and the used of machine and labor was considered. The CEMs were corrected after all the errors were collected. Figure 2.8 shows the spreadsheet-testing page for the Pultrusion process. All other processes have the same structure for the spreadsheets.

Usability testing

To discover the problems that occur during usage, a test group tested out the CEMs. The test group consisted of three MIT graduate students who all were research assistants in the composites field. The three students understand the Hand Layup process well, but did not know as much about the other processes. Thus, they represent the expert user (Hand Layup) and the novice user for the ATP, Forming, Pultrusion, and RTM processes. The outcome was extremely satisfactory. The users understood, enjoyed and showed interest in using the CEMs once published in the public domain. Having used the Internet as a tool for material selection, they think that it is a very useful technology. However, they have not seen any web models as in CEMs. This gave the research group much confidence that people in composites industry will actually use the CEMs. All the suggestions from a test group are incorporated into the final CEMs.

Process	Pultrusion
Shape	Flat (C1)

Dimension	
Width	4
Length	10
Thickness	0.125

mcost	18		
mrate	56	Ply Thickness	0.007
X	0.144642857	Number of Ply	18 Nearest Integer

Quantity	1

Assume: Setup for every part you make!!!

Process Plan									
Steps		Process Time	(Minute)	Labor Time	Machine Time	Y/N (1 / 0) (check)	Variable	Num()	ID
	Non-Recurring	Recurring	Total						
Machine Setup									
Identify Required Items for Pultrusion	8.0	0.0	8.0	8.0	0	1	0	1	550
Setup Pultrusion Equipment	130.0	0.0	130.0	130.0	130	1	0	1	2070
Setup Inline Ultrasonic Inspection Equipment	24.0	0.0	24.0	24.0	24	1	0	1	2020
Load Form Die onto Pultrusion Equipment	70.0	0.0	70.0	140.0	70	1	0	1	1140
Attach Thermocouple lines to Pultrusion Die	1.0	10.0	11.0	11.0	11	1	0	2	120
Total	233.0	10.0	243.0	313.0	235				
Material Setup									
Load Dry Fiber Material onto Pultrusion Equipment	60.0	0.0	60.0	120.0	60	1	0	1	900
Total	60.0	0.0	60.0	120.0	60				
Setup Resin Bath									
Setup Resin Bath	0.0	0.0	0.0	0.0	0	1	0	1	0
Load Resin into Resin Bath	0.0	0.0	0.0	0.0	0	1	0	1	0
Total	0	0.0	0.0	0.0	0				
Setup Resin Injection Machine									
Setup Resin Injection Machine	5.0	0.0	5.0	5.0	5	1	0	1	2080
Load Two Part Resin onto Pultrusion Injection Machine	0.0	0.0	0.0	0.0	0.006515444	1	0.144642857	1	930
Attach Resin Injection Lines to Pultrusion Die	2.0	10.0	12.0	12.0	12	1	0	1	100
Total	7.0	10.0	17.0	17.0	17.00651544				
Pultrusion									
Run Pultrusion Equipment	9.0	36.0	45.0	45.0	45	1	0.125	1	3000
Remove Form Die from Pultrusion Equipment	70.0	0.0	70.0	140.0	70	1	0	1	1660
Clean Pultrusion Form Die	10.0	2.6	12.6	12.6	12.6	1	4	1	210
Total	89.0	38.6	127.6	197.6	127.6				
Finishing									
Setup NC Trimming Equipment	15.0	0.0	15.0	15.0	15.0	1	0	1	2040
Position Part into NC Trimming Equipment	1.0	2.1	3.1	3.1	3.1	1	10	1	1210
Trim Automated Edge gr/ep	4.0	2.2	6.2	6.2	6.2	1	10	2	2220
Remove Finished Part from NC Trimming Equipment	1.0	1.3	2.3	2.3	2.3	1	10	1	1730
Manual Deburr Edge	8.0	0.9	8.9	8.9	0.0	1	10	2	2340
Total	29.0	6.5	35.5	35.5	26.6				
Inspection									
Setup Offline NC Ultrasonic Equipment	0.0	0.0	0.0	0.0	0.0	0	0	1	2030
Position Part into NC Ultrasonice Inspection Equipment	0.0	0.0	0.0	0.0	0.0	0	10	1	1000
Ultrasonic Inspect Part in NC Ultrasonic Equipment	0.0	0.0	0.0	0.0	0.0	0	10	1	570
Remove Finished Part from NC Ultrasonic Equipment	0.0	0.0	0.0	0.0	0.0	0	10	1	1730
Total	0.0	0.0	0.0	0.0	0.0				

Step / Time	Process Time	Labor Time	Labor Rate	Machine Time
Machine Setup	243.0	313.0		235.0
Material Setup	60.0	120.0		60.0
Setup Resin Bath	0.0	0.0		0.0
Setup Resin Injection Machine	17.0	17.0		17.0
Pultrusion	127.6	197.6		127.6
Finishing	35.5	35.5		26.6
Inspection	0.0	0.0		0.0
Total	483.1	683.1		466.2

Figure 2.8 Spreadsheet for the Pultrusion Process

Publishing to the WEB

The CEMs are published to our research group site @

http://web.mit.edu/lmp/www/composites/costmodel/

Any future updates will be posted to this location.

2.4 Equations

The process step Equation#1, Equation#2, Equation#5, and Equation#6 were incorporated into a framework which facilitates rapid estimates of different factory and design configurations, by [Neoh]. Equation#7 and Equation#10 were developed by [Haffner, 4,5]. Equation#8 was developed by [Beresheim]. The list of used equations includes:

Equation#1

$$standard\ minutes = \#\ setup \times [setup + (\#\ part\ per\ setup \times \#\ run \times delay)]$$

Equation#2

$$standard\ minutes = \#\ setup \times [setup + \#\ part\ per\ setup \times \#\ run \times (delay + \sqrt{(v1/vo1)^2 + (2.\tau 1.v1/vo1)})]$$

Equation#5

$$standard\ minutes = \#\ setup \times [setup + (\#\ part\ per\ setup \times \#\ run \times (delay + (v1/vo1)))]$$

Equation#6

$$standard\ minutes = \#\ setup \times [setup + \#\ part\ per\ setup \times \#\ run \times (delay + \sqrt{(v1/vo1)^2 + (2.\tau 1.v1/vo1)}) \times v2]$$

Equation#7

$$standard\ minutes = \#\ setup \times [setup + (\#\ part\ per\ setup \times \#\ run \times (delay / v))]$$

Equation#8

$$standard\ minutes = \#\ setup \times \#\ part\ per\ setup \times \#\ ply \times [A/(w.Vo) + \sum Nce \times (tce + 2.ta) + ns \times (tdelay + 2.ta')]$$

Equation#10

$$s\tan dard\ \min utes = \#\ setup \times [setup + \{\#\ part\ per\ setup \times \#\ run \times \#\ ply \times$$
$$(delay + T\sin g \times \sqrt{[\{(A\sin g/(V\sin g.T\sin g)) + 1\}^2 - 1]} + \tau 1 \times \sqrt{[\{(Adouble/(Vdouble.\tau 1)) + 1\}^2 - 1]}\}]$$

The estimated variable, number of runs, and boolean of machines used are passed to the equations at the time calculation function. These become the tedious part of the CEM because each step is different from the others, thus I need to go through every steps, reasoning case by case. T two examples of variations in the CEM are here given:

1) For the Hand Layup process of the flat panel part, the material setup main step includes of the cut prepreg step (Process ID = 2280) and the cut vacuum bag step (Process ID = 2280). Both have the same Process ID, but each is estimated differently. The process step is trim manual edge, and is estimated by the trim length. For the cut prepreg step, the trim length equals the prepreg perimeter with the assumption of two inches increment from all sides of the part. Even with the same Process ID, the trim length of the cut vacuum bag step has to be vacuum perimeter which is assumed to have a four inch increment from all part sides.
2) For the Pulrusion process of straight T-channel, the finishing main step includes manual debur edge step (Process ID = 2340) and position part into NC trimming equipment step (Process ID = 1210). Both are estimated with length, but they differ in the number of runs. For the manual debur edge, the number of runs is three since it is deburred on the three lengths of the T-channel. However for the position part into NC trimming equipment, the number of runs is only one because in positioning the part one only needs to position with the length of one side.

Chapter 3 Program Details

This chapter aims to show the details of the Cost Estimation Model (CEM). Since the overall structure has already been explained in Chapter 2, this chapter shows how the CEM works in a more detailed fashion. All the processes (Hand Layup, ATP, RTM, Pultrusion, and Forming), for making the 14 shapes, have the same structure. Although Hand Layup has been selected as the example to give insight into the model, one can apply these same sequences to all the other processes.

One simple curved part, (C2), is to be produced, using this Hand Layup process. The step by step procedure explains what should be inputted, how the calculation should be executed, how the model can be used in making the decisions, and how the user can play around with this model.

Figure 3.1, the flow diagram of the model, shows the sequences of the execution, the ordering of the values to be inputted, the calling up of the values from the databases, and the calculations to be made.

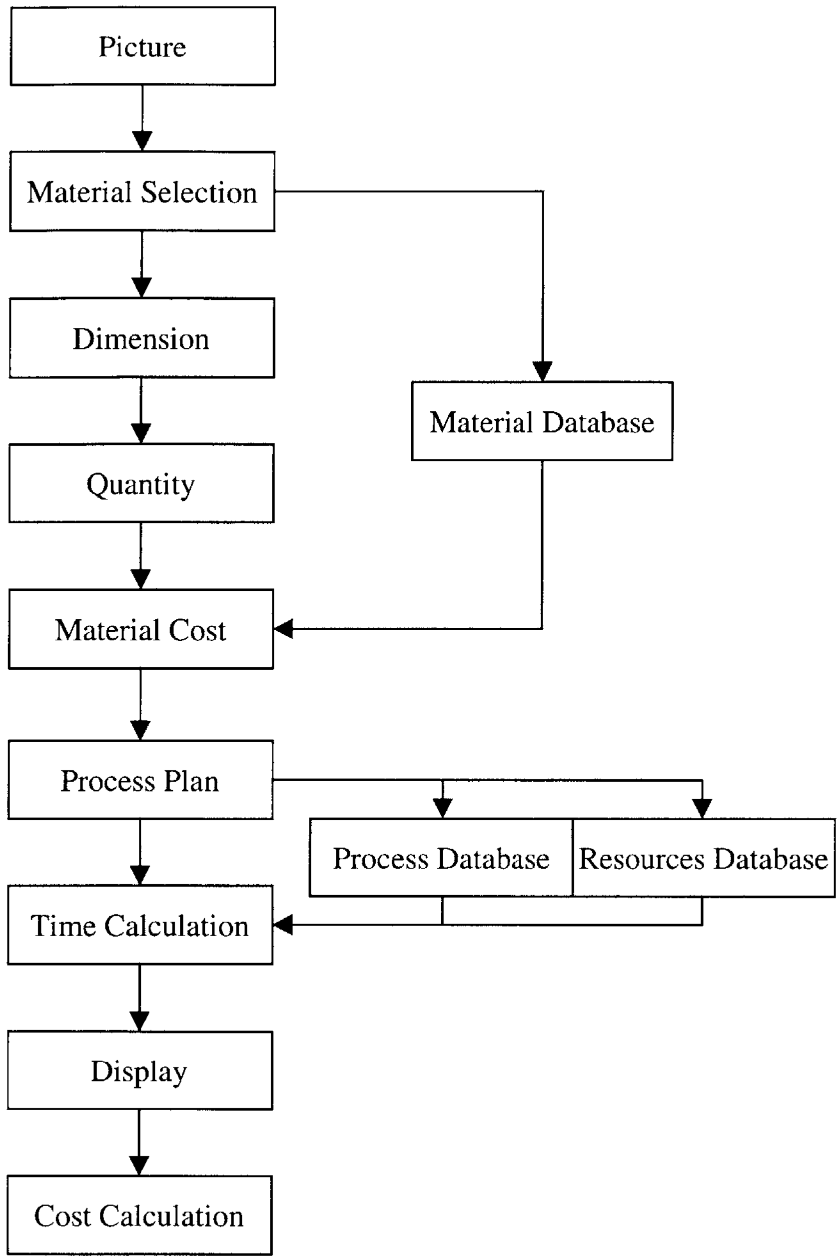


Figure 3.1 Flow Diagram of the CEM

The Cost Estimation Model (CEM) is broken into the following nine sections, as seen in the flow diagram sequences:

Picture

Material selection

Dimensioning

Quantity

Material cost

Process plan

Time calculation

Display

Cost calculation

3.1 Picture

The picture of the particular shape is shown on the left. In this case, a simple curved part (C2) is shown. The user has to refer to this picture to the dimension of the part to be manufactured. Please refer to Figure 3.2.

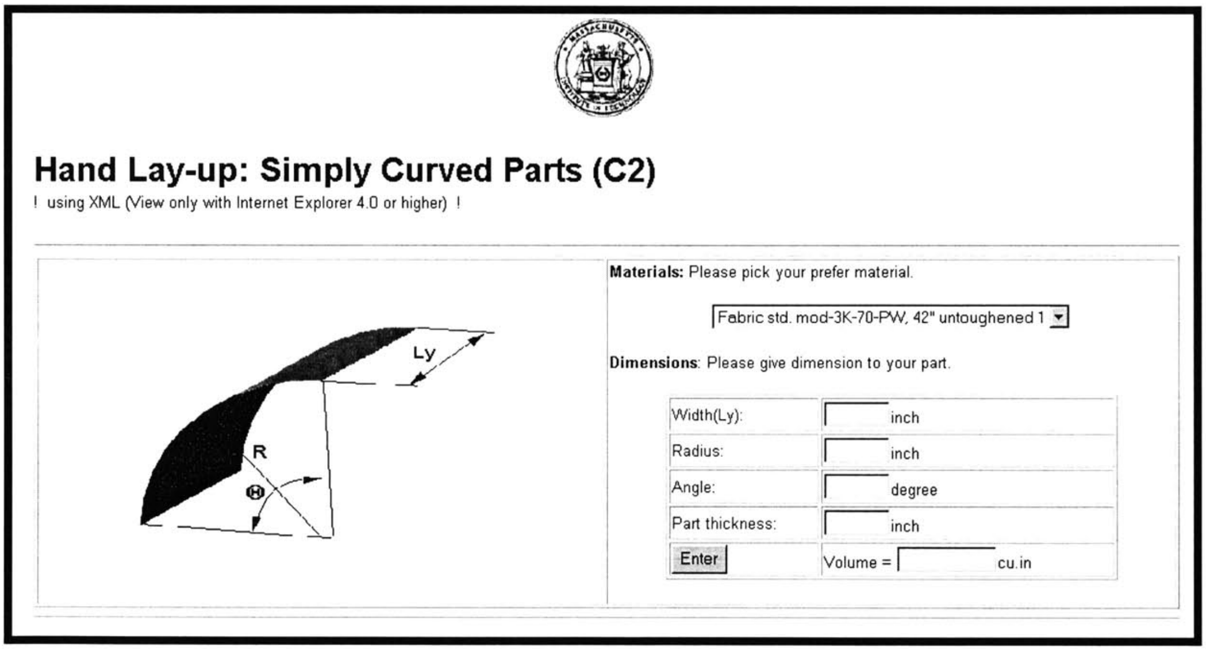


Figure 3.2 Picture of the Selected Shape

3.2 Material Selection

The first step in modeling of the process is the selection of the material. In any industry, the designer will usually have particular materials which can be used for certain processes in mind. Then, s/he will usually look at different materials and pick select the best one for production. Because not all materials can be used for a particular process, the user first needs to have some understanding about the material. Currently, the materials' listing is collected from the PCAD data. Although the material properties (density) are still accurate, other material information (width, price, thickness, and typical scrap) may be a little dated. An undergraduate student is presently conducting research to update these informational aspects. At the completion of this project, the new material database will be updated for the CEM. Figure 3.3 shows the material selection list. For this example, Fabric Std. Mod-3K-70-PW, 42" untoughened 1 is selected. With the material selected, JavaScript calls up the Material XML database which then feeds the information back to the model. This information includes the material name, material ID, density (lb/in^3), price ($/lb), typical scrap rate (%), thickness (inch), and width (inch).

Our research group goal is to work with companies in composites industry so that with the exchange of information, both the companies and our research group will benefit from each other, thus gaining better and more accurate information on the database.

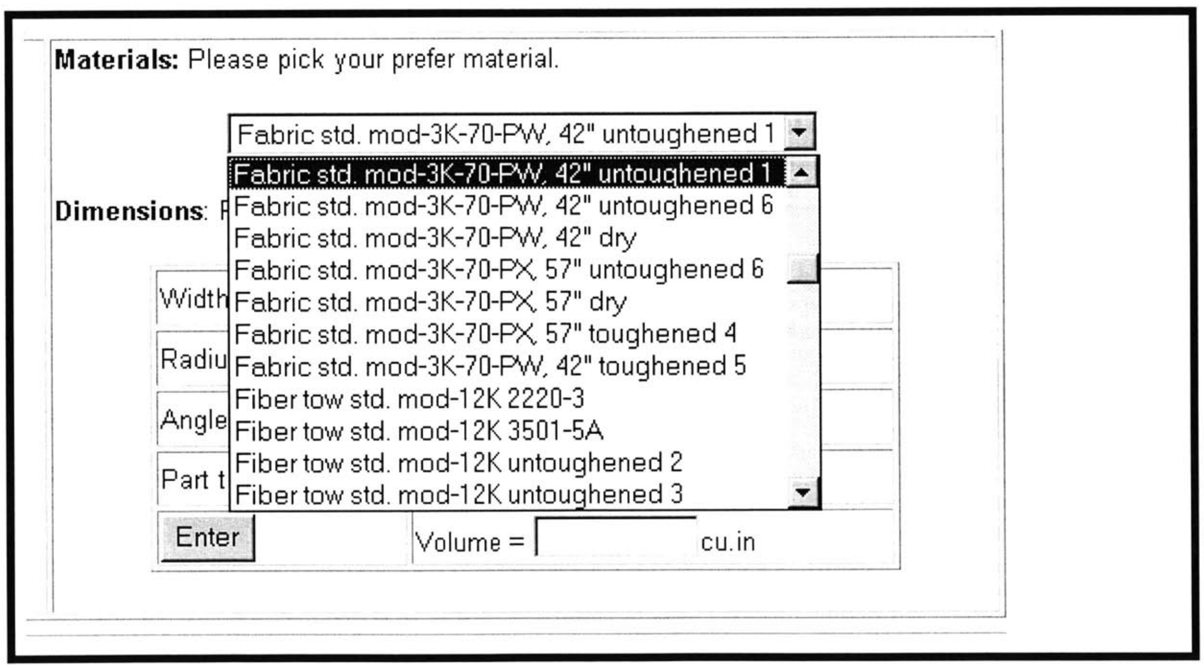


Figure 3.3 Material Selection

3.3 Dimensioning

Base on the picture on the left side of the input box, the user can input the dimension of the part to be processed. All dimensions have units, expressed on the right of the input boxes. For length, all units are in inches. For angle, all units are in degrees. After the values have been inputted, user has to click "Enter" to calculate and display the volume.

This button does not only calculate the volume, but it also gets the values from database and inputs all information for subsequent calculations. Thus, user must click this button, in order to go on to the next sequence. All the inputs are written in a blue text, whereas a black text is for the values not to be inputted.

For our example of a simple curved part for Hand Layup, the dimensioning is:

Width (Ly) = 60 inches

Radius = 46 inches

Angle = 30 degrees

Part Thickness = 0.2 inch

Volume = width x arc length x part thickness

= 60 x (46 x 30 x ¶ / 180) x 0.2

= 289 in^3

This dimensioning can be seen in Figure 3.4.

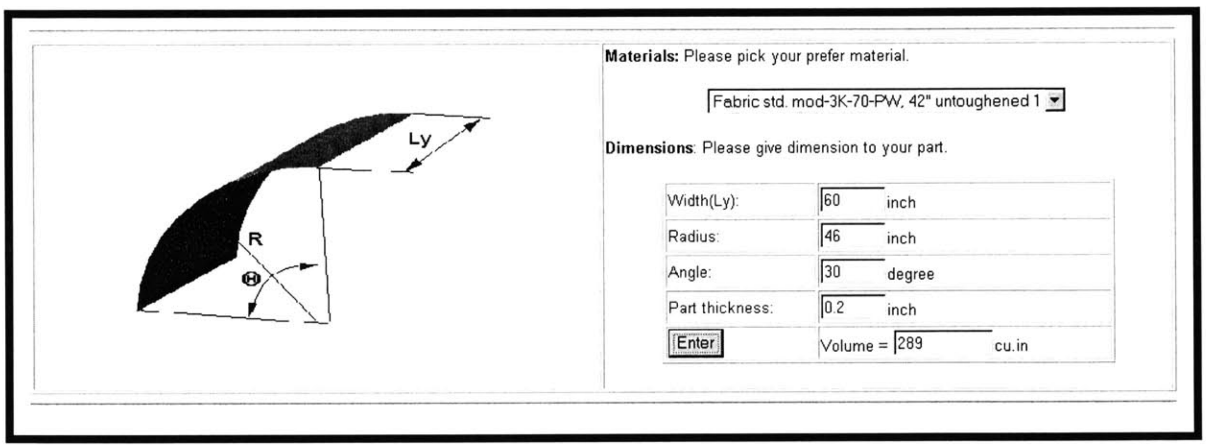


Figure 3.4 Dimension of the Shape (Hand Layup of Simple Curved Part)

3.4 Quantity

In this section, the quantity of parts to be produced is inputted. The user needs to input both the parts made per setup and the number of setup. The calculations will vary greatly, depending on these values, as discussed in Chapter 2. The default value for the number of setup of 1 is given. User can overwrite this default value. Figure 3.5 shows the quantity section.

For this particular example, the values inputted are:

Parts made per setup = 1

Number of setups = 1

Total quantity = part made per setup x number of setups

= 1 x 1 = 1 part

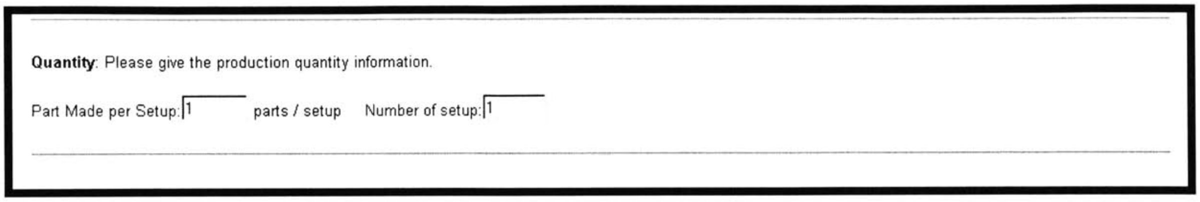


Figure 3.5 Quantity Input

3.5 Material Cost

This section includes all the details used in further calculations, including both material and time calculations. The default values are given to each input box from the database. Again, the user can input specific material information for particular material used by any particular company or its suppliers.

The information includes in this section is material rate, thickness, density, width, and scrap. For our example of simple curved part, the material has:

Material rate = $47.5 / lb.

Thickness = 0.007 in

Density = 0.057 lb/in^3

Width = 42 inches

Scrap = 15 %

The user is then required to click "Calculate Material Cost" to calculate the material cost and use the entered information in later calculations. Also, if the user wants to see the entire material database, s/he can click "View Database" to see the XSL database. The material cost section is shown in Figure 3.6. In this example, the material cost is calculated to be $900.

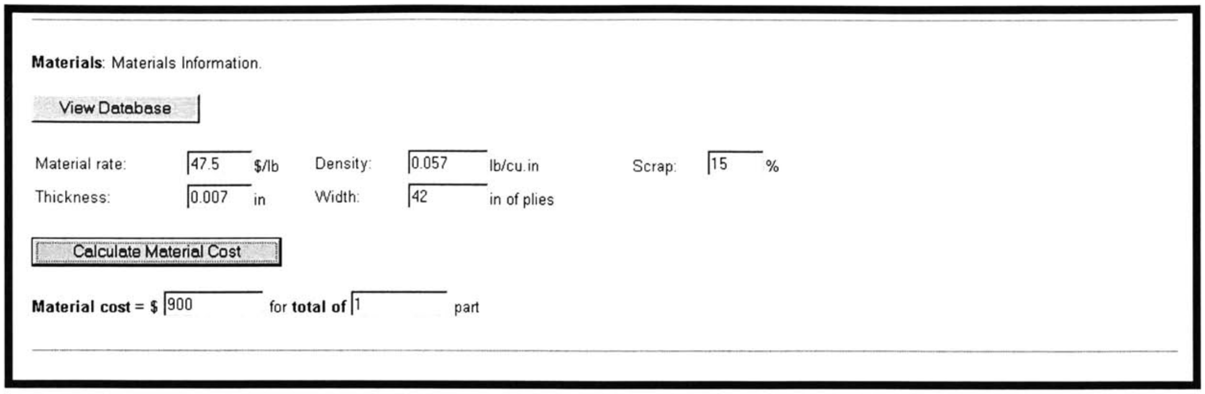


Figure 3.6 Material Information and Cost Calculation

3.6 Process Plan

This is the most important part of the entire CEM because all the main calculations occur here. It is essential that each step which is to be included be "CHECKED". As a result, the steps that are "UNCHECKED", will neither be included nor calculated. The default plan checks all the steps included in the typical process plan. Moreover, up to four user additional steps can be included. Once the user has checked and unchecked all the steps, calculations can then be performed. Figure 3.7 shows the "CHECKED" and "UNCHECKED" whereas Figure 3.8 shows the structure of the process plan. The process plan contains various main steps, and each main step consisting of many steps. Figure 4.9 shows the default process plan for Hand Layup. Note that the Process and Resources databases are called by the checked items.

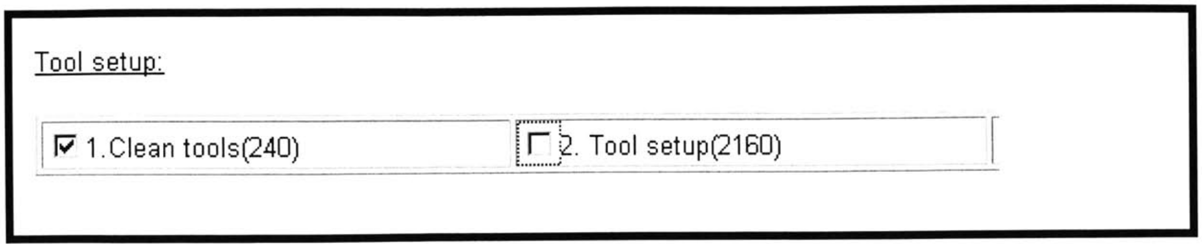


Figure 3.7 Step Selection of "CHECKED" and "UNCHECKED"

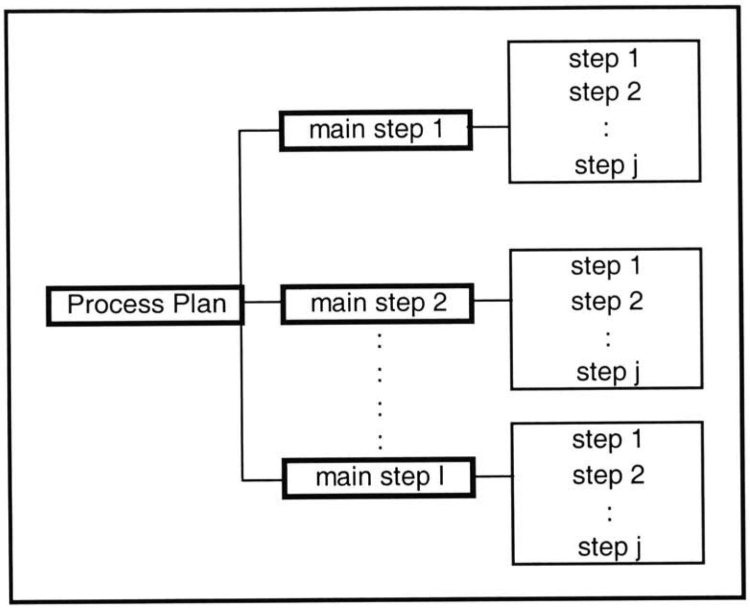


Figure 3.8 Structure of the Process Plan

User can click "View Database" to see all the process parameters for each of the process steps. If the user wishes to see only one step at a time, s/he can click "Individual Parameters" with very easy-to-use results. The user can type in the process ID and click "Enter" to find out about all information used in the time calculation.

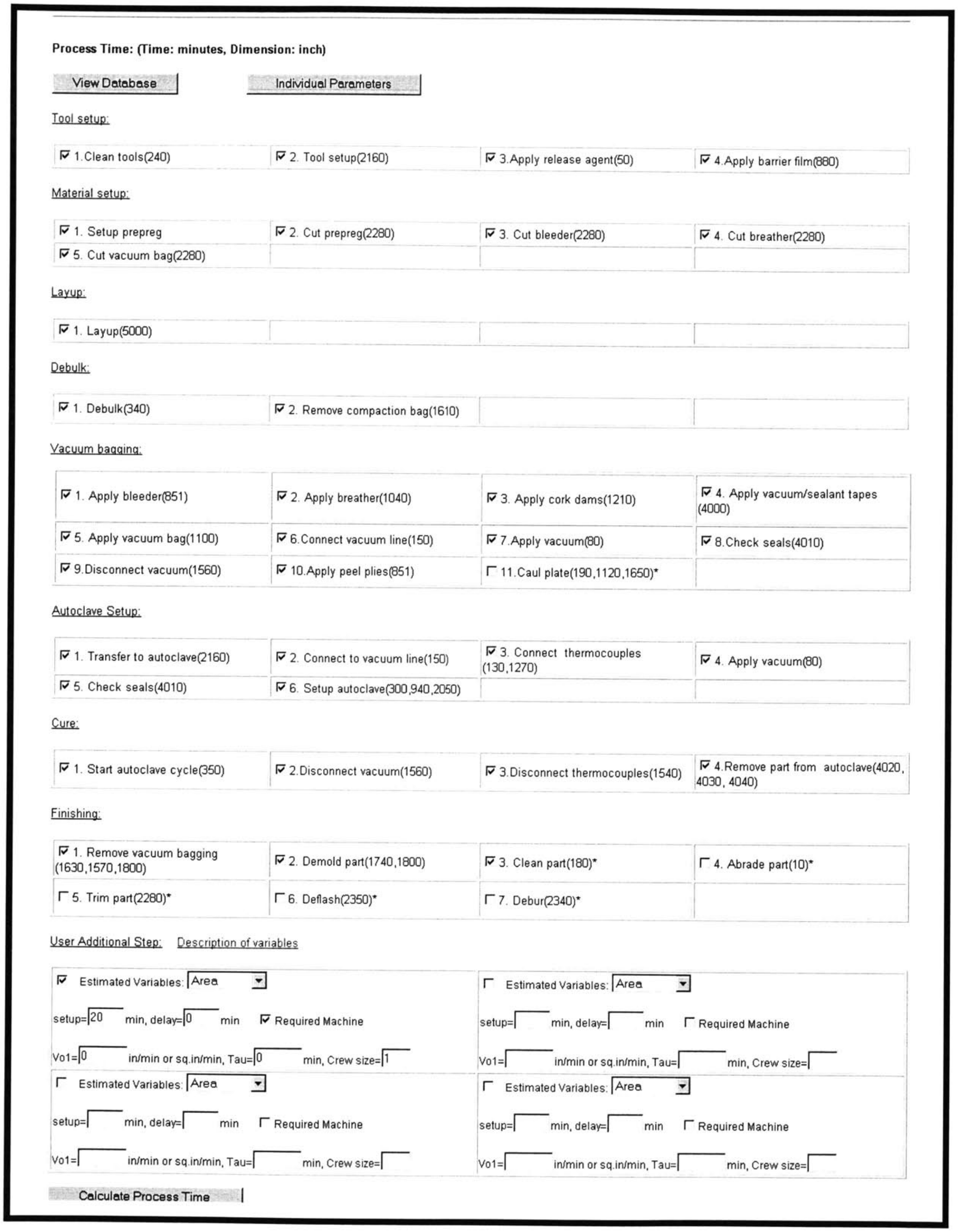


Figure 3.9 Process Plan (Hand Layup Process)

3.7 Time Calculation

Once "Calculate Process Time" is clicked, all calculations are performed. Both the process and resources databases are called, then. the values are passed on to JavaScript calculations.

The parameters to be used in the calculations are:

General parameters

Number of ply = n = round up (part thickness / ply thickness) = 0.2 / 0.007 = 26 plies

Part area = arc length x width = 24.1 x 60 = 1446 in^2

Ply area = Inclusion of 2 inches on each side to part area

= 28.1 x 64 = 1798.4 in^2

Vacuum bag area = Inclusion of 2 inches on each side to ply area

= 32.1 x 68 = 2182.8 in^2

Part perimeter = 2 x (arc length + width) = 2 x (24.1 + 60) = 168.2 in

Ply perimeter = 2 x (28.1 + 64) = 184.2 in

Vacuum bag perimeter = 2 x (32.1 + 68) = 200.2 in

Width = 60 in

Layup parameters

I = delta x width = (30 x ¶ / 180) x 60 = 31.4 in

$\Delta\Theta g = 0$ rad

Area, single = part area = 1446 in^2

Area, double = 0 in^2

The equations to be used in the Hand Layup example are:

Labor time = crew size x process time

Machine time = process time of steps with the use of the machine

Equation1

$$standard\ minutes = \#\ setup \times [setup + (\#\ part\ per\ setup \times \#\ run \times delay)]$$

Equation2

$$standard\ minutes = \#\ setup \times [setup + \#\ part\ per\ setup \times \#\ run \times (delay + \sqrt{(v1 / vo1)^2 + (2.\tau 1.v1 / vo1)})]$$

Equation5

$$standard\ minutes = \#\ setup \times [setup + (\#\ part\ per\ setup \times \#\ run \times (delay + (v1 / vo1)))]$$

Equation6

$$standard\ minutes = \#\ setup \times [setup + \#\ part\ per\ setup \times \#\ run \times (delay + \sqrt{(v1/vo1)^2 + (2.\tau 1.v1/vo1)}) \times v2]$$

Equation10 [Haffner, 5]

Vsing = Vo1 / (1 + (Vo1.I/Cn))

Vdouble = Vo1 / (1 + (Vo1ΔΘg /Cg))

If (radius ≥ 12), then Tsing =τ1

If (radius < 12), then Tsing = τ1 + (bn.ΔΘ.width)

$$standard\ minutes = \#\ setup \times [setup + \{\#\ part\ per\ setup \times \#\ run \times \#\ ply \times (delay + T\sin g \times \sqrt{[\{(A\sin g/(V\sin g.T\sin g)) + 1\}^2 - 1]} + \tau 1 \times \sqrt{[\{(Adouble/(Vdouble.\tau 1)) + 1\}^2 - 1]}\}]$$

Note:

\# of setups = bps = batch per setup

\# part per setup = ppb = part per batch

\#run = number of run

For the Hand Layup of simple curved part, the calculations are as follows:

Tool setup

Clean tools (240)

V1 = vacuum bag area, #run = 1, machine required = "No", Equation2

$$Process\ time = 1 \times [3 + \{1 \times 1 \times (0 + \sqrt{(2182.8/493.34)^2 + (2 \times 7.105 \times 2182.8/493.34)})\}]$$

Process time = 12 min (3 non-recurring, 9 recurring)

Labor time = 1x process time = 12 min

Machine time = 0 min

Tool setup (2160)

V1 = 0, #run = 1, machine required = "No", Equation1

$$Process\ time = 1 \times [34.6 + (1 \times 1 \times 0)]$$

Process time = 35 min (35 non-recurring, 0 recurring)

Labor time = 2 x process time = 69 min

Machine time = 0 min

Apply release agent (50)

V1 = vacuum bag area, #run = 1, machine required = "No", Equation2

$$Process\ time = 1 \times [5 + \{1 \times 1 \times (0 + \sqrt{(2182.8/612)^2 + (2 \times 2.25 \times 2182.8/612)})\}]$$

Process time = 10 min (5 non-recurring, 5 recurring)

Labor time = 1x process time = 10 min

Machine time = 0 min

Apply barrier film (880)

V1 = ply area, #run = 1, machine required = "No", Equation6, V2 = 1

$$Process\ time = 1 \times [1 + \{1 \times 1 \times (2 + \sqrt{(1798.4/2500)^2 + (2 \times 0 \times 1798.4/2500)}) \times 1\}]$$

Process time = 4 min (1 non-recurring, 3 recurring)

Labor time = 2 x process time = 7 min

Machine time = 0 min

<u>Material setup</u>

Setup prepreg (none)

Not "Checked".

Process time, labor time, and machine time = 0 min

Cut prepreg (2280)

V1 = ply perimeter, #run = 1, machine required = "No", Equation2

$$Process\ time = 1 \times [6 + \{1 \times 1 \times (1 + \sqrt{(184.2/30)^2 + (2 \times 2 \times 184.2/30)})\}]$$

Process time = 15 min (6 non-recurring, 9 recurring)

Labor time = 1x process time = 15 min

Machine time = 0 min

Cut bleeder (2280)

V1 = ply perimeter, #run = 1, machine required = "No", Equation2

$$Process\ time = 1\times[6+\{1\times1\times(1+\sqrt{(184.2/30)^2+(2\times2\times184.2/30)})\}]$$

Process time = 15 min (6 non-recurring, 9 recurring)

Labor time = 1x process time = 15 min

Machine time = 0 min

Cut breather (2280)

V1 = ply perimeter, #run = 1, machine required = "No", Equation2

$$Process\ time = 1\times[6+\{1\times1\times(1+\sqrt{(184.2/30)^2+(2\times2\times184.2/30)})\}]$$

Process time = 15 min (6 non-recurring, 9 recurring)

Labor time = 1x process time = 15 min

Machine time = 0 min

Cut vacuum bagging (2280)

V1 = vacuum bag perimeter, #run = 1, machine required = "No", Equation2

$$Process\ time = 1\times[6+\{1\times1\times(1+\sqrt{(200.2/30)^2+(2\times2\times200.2/30)})\}]$$

Process time = 13 min (6 non-recurring, 7 recurring)

Labor time = 1x process time = 13 min

Machine time = 0 min

<u>Layup</u>

Layup (5000)

V1 = part area, #run = 1, machine required = "No", Equation10

Radius $\geq$ 12, thus

Vsing = 935 / (1 + (935x 31.4 / 7250000)) = 931.23 in^2/min

Vdouble = 935 / (1 + (935 x 0 / 36.39)) = 935 in^2/min

$$\Pr ocess\ time = 1 \times [0 + \{1 \times 1 \times 26 \times$$
$$(0 + 5.814 \times \sqrt{[\{(1446/(921.23 \times 5.814)) + 1\}^2 - 1]} + 5.814 \times \sqrt{[\{(0/(935 \times 5.814)) + 1\}^2 - 1]}\}]$$

Process time = 131 min (0 non-recurring, 131 recurring)

Labor time = 1x process time = 131 min

Machine time = 0 min

Debulk every 2 plies

Debulk (340)

V1 = 0, #run = 13, machine required = "No", Equation1

$$\Pr ocess\ time = 1 \times [9 + (1 \times 13 \times 0)]$$

Process time = 9 min (9 non-recurring, 0 recurring)

Labor time = 1 x process time = 9 min

Machine time = 0 min

Remove compaction bag (1610)

V1 = vacuum bag perimeter, #run = 1, machine required = "No", Equation2

$$\Pr ocess\ time = 1 \times [0.6 + \{1 \times 13 \times (0 + \sqrt{(200.2/28.893)^2 + (2 \times 2.465 \times 200.2/28.893))}\}]$$

Process time = 94 min (1 non-recurring, 94 recurring)

Labor time = 2 x process time = 188 min

Machine time = 0 min

Vacuum bagging

Apply bleeder (851)

V1 = ply area, #run = 1, machine required = "No", Equation2

$$\Pr ocess\ time = 1 \times [4 + \{1 \times 1 \times (1 + \sqrt{(1798.4/347.63)^2 + (2 \times 0 \times 1798.4/347.63))}\}]$$

Process time = 10 min (4 non-recurring, 6 recurring)

Labor time = 2 x process time = 20 min

Machine time = 0 min

Apply breather (1040)

V1 = ply perimeter, #run = 1, machine required = "No", Equation2

$$\Pr ocess\ time = 1 \times [2 + \{1 \times 1 \times (0 + \sqrt{(184.2/30)^2 + (2 \times 1.22 \times 184.2/30)})\}]$$

Process time = 9 min (2 non-recurring, 7 recurring)

Labor time = 1 x process time = 9 min

Machine time = 0 min

Apply cork dams (1210)

V1 = vacuum bag perimeter, #run = 1, machine required = "No", Equation2

$$\Pr ocess\ time = 1 \times [1 + \{1 \times 1 \times (1 + \sqrt{(200.2/9.2)^2 + (2 \times 0 \times 200.2/9.2)})\}]$$

Process time = 22 min (1 non-recurring, 21 recurring)

Labor time = 1 x process time = 22 min

Machine time = 0 min

Apply vacuum/sealant tapes (4000)

V1 = vacuum bag perimeter, #run = 1, machine required = "No", Equation5

$$\Pr ocess\ time = 1 \times [0 + (1 \times 1 \times (0 + (200.2/30.86)))]$$

Process time = 4 min (0 non-recurring, 4 recurring)

Labor time = 1 x process time = 4 min

Machine time = 0 min

Apply vacuum bag (1100)

V1 = vacuum bag area, #run = 1, machine required = "No", Equation2

$$\Pr ocess\ time = 1 \times [5 + \{1 \times 1 \times (0 + \sqrt{(2182.8/600)^2 + (2 \times 1.4 \times 2182.2/600)})\}]$$

Process time = 10 min (5 non-recurring, 5 recurring)

Labor time = 2 x process time = 20 min

Machine time = 0 min

Connect vacuum line (150)

V1 = vacuum bag perimeter, #run = 1, machine required = "No", Equation2

$$Process\ time = 1 \times [1 + \{1 \times 1 \times (1 + \sqrt{(200.2/200)^2 + (2 \times 0 \times 200.2/200)})\}]$$

Process time = 3 min (1 non-recurring, 2 recurring)

Labor time = 1 x process time = 3 min

Machine time = 0 min

Apply vacuum (80)

V1 = 0, #run = 1, machine required = "No", Equation1

$$Process\ time = 1 \times [5 + (1 \times 1 \times 10)]$$

Process time = 15 min (5 non-recurring, 10 recurring)

Labor time = 1 x process time = 15 min

Machine time = 0 min

Check seals (4010)

V1 = vacuum bag perimeter, #run = 1, machine required = "No", Equation5

$$Process\ time = 1 \times [0 + (1 \times 1 \times (0 + (200.2/980.4)))]$$

Process time = 0 min (0 non-recurring, 0 recurring)

Labor time = 1 x process time = 0 min

Machine time = 0 min

Disconnect vacuum (1560)

V1 = vacuum bag perimeter, #run = 1, machine required = "No", Equation2

$$Process\ time = 1 \times [1 + \{1 \times 1 \times (0.5 + \sqrt{(200.2/400)^2 + (2 \times 0 \times 200.2/400)})\}]$$

Process time = 2 min (1 non-recurring, 1 recurring)

Labor time = 1 x process time = 2 min

Machine time = 0 min

Apply peel plies (851)

V1 = ply area, #run = 1, machine required = "No", Equation2

$$Process\ time = 1 \times [4 + \{1 \times 1 \times (1 + \sqrt{(1798.4/347.63)^2 + (2 \times 0 \times 1798.4/347.63)})\}]$$

Process time = 10 min (4 non-recurring, 6 recurring)

Labor time = 2 x process time = 20 min

Machine time = 0 min

Caul plate (190, 1120, 1650)

Not "Checked".

Process time, labor time, and machine time = 0 min

<u>Autoclave setup</u>

Transfer to autoclave (2160)

V1 = 0, #run = 1, machine required = "Yes", Equation1

$$Process\ time = 1 \times [34.6 + (1 \times 1 \times 0)]$$

Process time = 35 min (35 non-recurring, 0 recurring)

Labor time = 1 x process time = 35 min

Machine time = 35 min

Connect to vacuum line (150)

V1 = part perimeter, #run = 1, machine required = "Yes", Equation2

$$Process\ time = 1 \times [1 + \{1 \times 1 \times (1 + \sqrt{(168.2/200)^2 + (2 \times 0 \times 168.2/200)})\}]$$

Process time = 3 min (1 non-recurring, 2 recurring)

Labor time = 1 x process time = 3 min

Machine time = 3 min

Connect thermocouples (130, 1270)

130: Attach lines

V1 = part perimeter, #run = 1, machine required = "Yes", Equation2

$$Process\ time = 1 \times [2 + \{1 \times 1 \times (1 + \sqrt{(168.2/27)^2 + (2 \times 0 \times 168.2/27)})\}]$$

Process time = 9 min (2 non-recurring, 7 recurring)

Labor time = 1 x process time = 9 min

Machine time = 9 min

1270: Position

V1 = part perimeter, #run = 1, machine required = "Yes", Equation2

$$\Pr ocess\ time = 1\times[2+\{1\times1\times(0+\sqrt{(168.2/27)^2+(2\times0\times168.2/27)})\}]$$

Process time = 8 min (2 non-recurring, 6 recurring)

Labor time = 1 x process time =8 min

Machine time = 8 min

Apply vacuum (80)

V1 = 0, #run = 1, machine required = "Yes", Equation1

$$\Pr ocess\ time = 1\times[5+(1\times1\times10)]$$

Process time = 15 min (5 non-recurring, 10 recurring)

Labor time = 1 x process time = 15 min

Machine time = 0 min

Check seals (4010)

V1 = vacuum bag perimeter, #run = 1, machine required = "Yes", Equation5

$$\Pr ocess\ time = 1\times[0+(1\times1\times(0+(200.2/980.4)))]$$

Process time = 0 min (0 non-recurring, 0 recurring)

Labor time = 1 x process time = 0 min

Machine time = 0 min

Setup autoclave (300, 940, 2050)

300: Close autoclave

V1 = 0, #run = 1, machine required = "Yes", Equation1

$$\Pr ocess\ time = 1\times[2+(1\times1\times12)]$$

Process time = 14 min (2 non-recurring, 12 recurring)

Labor time = 1 x process time = 14 min

Machine time = 14 min

940: Open autoclave

V1 = 0, #run = 1, machine required = "Yes", Equation1

$$\Pr ocess\ time = 1 \times [2 + (1 \times 1 \times 14)]$$

Process time = 16 min (2 non-recurring, 14 recurring)

Labor time = 1 x process time = 16 min

Machine time = 16 min

2050: Setup autoclave

V1 = 0, #run = 1, machine required = "Yes", Equation1

$$\Pr ocess\ time = 1 \times [16 + (1 \times 1 \times 0)]$$

Process time = 16 min (16 non-recurring, 0 recurring)

Labor time = 1 x process time = 16 min

Machine time = 16 min

Cure

Start autoclave cycle (350)

V1 = 0, #run = 1, machine required = "Yes", Equation1

$$\Pr ocess\ time = 1 \times [480 + (1 \times 1 \times 480)]$$

Process time = 480 min (480 non-recurring, 0 recurring)

Labor time = 1 x process time = 480 min

Machine time = 480 min

Disconnect vacuum (1560)

V1 = vacuum bag perimeter, #run = 1, machine required = "Yes", Equation2

$$\Pr ocess\ time = 1 \times [1 + \{1 \times 1 \times (0.5 + \sqrt{(200.2/400)^2 + (2 \times 0 \times 200.2/400)})\}]$$

Process time = 2 min (1 non-recurring, 1 recurring)

Labor time = 1 x process time = 2 min

Machine time = 2 min

Disconnect thermocouples (1540)

V1 = part perimeter, #run = 1, machine required = "Yes", Equation2

$$\Pr ocess\ time = 1 \times [1 + \{1 \times 1 \times (1 + \sqrt{(168.2/54)^2 + (2 \times 0 \times 168.2/54)})\}]$$

Process time = 5 min (1 non-recurring, 4 recurring)

Labor time = 1 x process time = 5 min

Machine time = 5 min

Remove part from autoclave (4020, 4030, 4040)

4020: Roll tray out

V1 = 0, #run = 1, machine required = "Yes", Equation1

$$\Pr ocess\ time = 1 \times [0.72 + (1 \times 1 \times 0)]$$

Process time = 1 min (1 non-recurring, 0 recurring)

Labor time = 1 x process time = 1 min

Machine time = 1 min

4030: Remove layup from tray

V1 = vacuum bag area, #run = 1, machine required = "Yes", Equation2

$$\Pr ocess\ time = 1 \times [0 + \{1 \times 1 \times (0 + \sqrt{(2182.8/3542.4)^2 + (2 \times 1.639 \times 2182.2/3542.4)})\}]$$

Process time = 1 min (0 non-recurring, 1 recurring)

Labor time = 1 x process time = 1 min

Machine time = 1 min

4040: Release clamps off tray

V1 = vacuum bag perimeter, #run = 1, machine required = "Yes", Equation5

$$\Pr ocess\ time = 1 \times [0 + (1 \times 1 \times (0 + (200.2/238.1)))]$$

Process time = 1 min (0 non-recurring, 1 recurring)

Labor time = 1 x process time = 1 min

Machine time = 1 min

<u>Finishing</u>

Remove vacuum bagging (1630, 1570, 1800)

1630: Remove cure bag

V1 = vacuum bag area, #run = 1, machine required = "No", Equation2

$$\Pr ocess\ time = 1 \times [5 + \{1 \times 1 \times (0 + \sqrt{(2182.8/1500)^2 + (2 \times 0.56 \times 2182.8/1500)})\}]$$

Process time = 7 min (5 non-recurring, 2 recurring)

Labor time = 2 x process time = 14 min

Machine time = 0 min

1570: Remove airweave

V1 = part perimeter, #run = 1, machine required = "No", Equation2

$$\Pr ocess\ time = 1 \times [1 + \{1 \times 1 \times (0 + \sqrt{(168.2/60)^2 + (2 \times 1.22 \times 168.2/60)})\}]$$

Process time = 5 min (1 non-recurring, 4 recurring)

Labor time = 1 x process time = 5 min

Machine time = 0 min

1800: Remove separation film

V1 = ply area, #run = 1, machine required = "No", Equation6, V2 = 1

$$\Pr ocess\ time = 1 \times [1 + \{1 \times 1 \times (2 + \sqrt{(1798.4/6250)^2 + (2 \times 0 \times 1798.4/6250)}) \times 1\}]$$

Process time = 3 min (1 non-recurring, 2 recurring)

Labor time = 1 x process time = 2 min

Machine time = 0 min

Demold part (1740, 1800)

1740: Remove peel ply

V1 = ply area, #run = 1, machine required = "No", Equation2

$$\Pr ocess\ time = 1 \times [1 + \{1 \times 1 \times (0 + \sqrt{(1798.4/142.87)^2 + (2 \times 0 \times 1798.4/142.87)})\}]$$

Process time = 14 min (1 non-recurring, 13 recurring)

Labor time = 1 x process time = 14 min

Machine time = 0 min

1800: Remove part

V1 = ply area, #run = 1, machine required = "No", Equation6, V2 = 1

$$\Pr ocess\ time = 1 \times [1 + \{1 \times 1 \times (2 + \sqrt{(1798.4/6250)^2 + (2 \times 0 \times 1798.4/6250)}) \times 1\}]$$

Process time = 3 min (1 non-recurring, 2 recurring)

Labor time = 1 x process time = 2 min

Machine time = 0 min

Clean parts (180)

V1 = part area, #run = 1, machine required = “No”, Equation2

$$\Pr ocess\ time = 1 \times [2.56 + \{1 \times 1 \times (0.5 + \sqrt{(1446/496.67)^2 + (2 \times 7.77 \times 1446/496.67)})\}]$$

Process time = 10 min (3 non-recurring, 8 recurring)

Labor time = 1 x process time = 10 min

Machine time = 0 min

Abrade part (10)

Not “Checked”.

Process time, labor time, and machine time = 0 min

Trim part (2280)

Not “Checked”.

Process time, labor time, and machine time = 0 min

Deflash (2350)

Not “Checked”.

Process time, labor time, and machine time = 0 min

Debur (2340)

Not “Checked”.

Process time, labor time, and machine time = 0 min

<u>User Additional</u>, (1 checked)

V1 = 0, #run = 1, machine required = “No”, Equation1

$$\Pr ocess\ time = 1 \times [20 + (1 \times 1 \times 0)]$$

Process time = 20 min (20 non-recurring, 0 recurring)

Labor time = 1 x process time = 20 min

Machine time = 0 min

Once the user clicks "Calculate Process Time", the time is automatically calculated. At this time, JavaScript calls XML databases (process and resources) to get the default labor and machine rates for each process step.

3.8 Display

This is probably the single most important part to the CEM user. S/he will be able to see all the time that has been calculated so far. Figure 3.10, showing the overall structure of the display, is broken down into three types:

- Process time
- Labor time
- Machine time

Each type is further broken down further into

- Non-recurring time which is the summation of setup time for all the steps selected.
- Recurring time which includes the delay time and operation time per run for the steps selected.

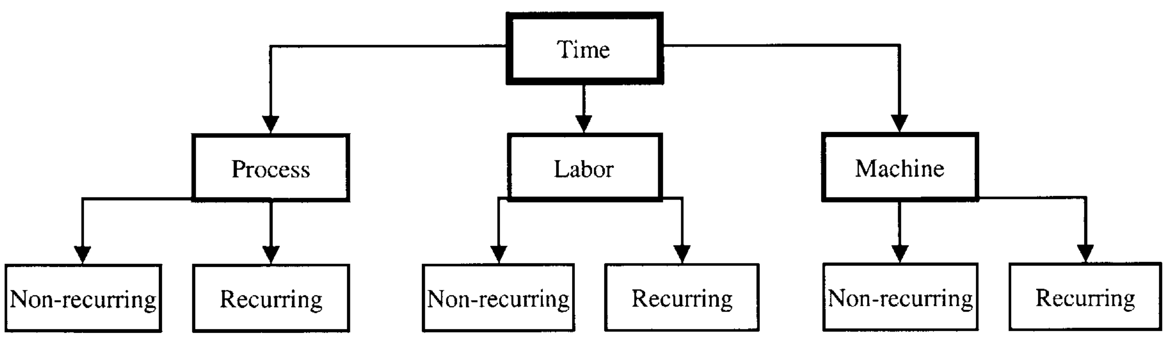
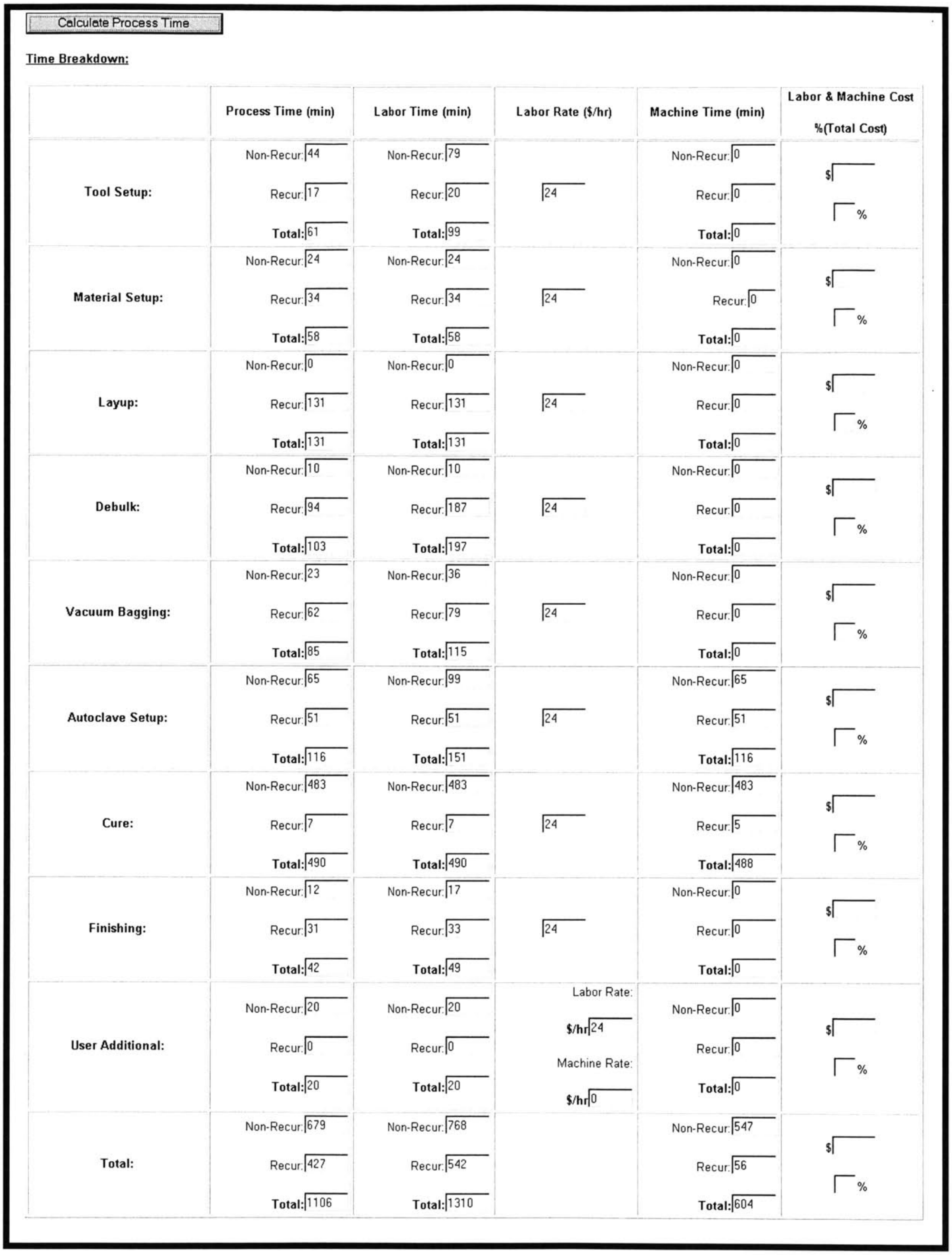


Figure 3.10 Time Structure

The display for this example is shown in Figure 3.11.



Calculate Process Time

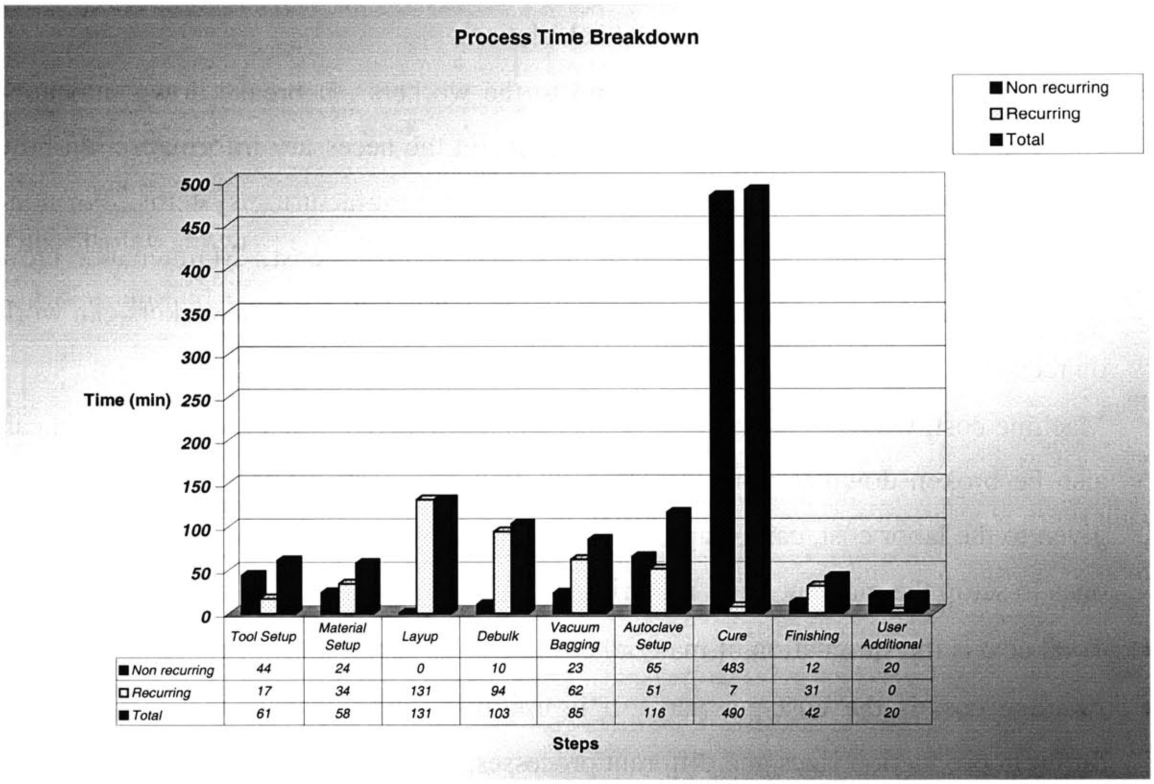
Time Breakdown:

	Process Time (min)	Labor Time (min)	Labor Rate ($/hr)	Machine Time (min)	Labor & Machine Cost %(Total Cost)
Tool Setup:	Non-Recur: 44 Recur: 17 **Total:** 61	Non-Recur: 79 Recur: 20 **Total:** 99	24	Non-Recur: 0 Recur: 0 **Total:** 0	$ %
Material Setup:	Non-Recur: 24 Recur: 34 **Total:** 58	Non-Recur: 24 Recur: 34 **Total:** 58	24	Non-Recur: 0 Recur: 0 **Total:** 0	$ %
Layup:	Non-Recur: 0 Recur: 131 **Total:** 131	Non-Recur: 0 Recur: 131 **Total:** 131	24	Non-Recur: 0 Recur: 0 **Total:** 0	$ %
Debulk:	Non-Recur: 10 Recur: 94 **Total:** 103	Non-Recur: 10 Recur: 187 **Total:** 197	24	Non-Recur: 0 Recur: 0 **Total:** 0	$ %
Vacuum Bagging:	Non-Recur: 23 Recur: 62 **Total:** 85	Non-Recur: 36 Recur: 79 **Total:** 115	24	Non-Recur: 0 Recur: 0 **Total:** 0	$ %
Autoclave Setup:	Non-Recur: 65 Recur: 51 **Total:** 116	Non-Recur: 99 Recur: 51 **Total:** 151	24	Non-Recur: 65 Recur: 51 **Total:** 116	$ %
Cure:	Non-Recur: 483 Recur: 7 **Total:** 490	Non-Recur: 483 Recur: 7 **Total:** 490	24	Non-Recur: 483 Recur: 5 **Total:** 488	$ %
Finishing:	Non-Recur: 12 Recur: 31 **Total:** 42	Non-Recur: 17 Recur: 33 **Total:** 49	24	Non-Recur: 0 Recur: 0 **Total:** 0	$ %
User Additional:	Non-Recur: 20 Recur: 0 **Total:** 20	Non-Recur: 20 Recur: 0 **Total:** 20	Labor Rate: **$/hr** 24 Machine Rate: **$/hr** 0	Non-Recur: 0 Recur: 0 **Total:** 0	$ %
Total:	Non-Recur: 679 Recur: 427 **Total:** 1106	Non-Recur: 768 Recur: 542 **Total:** 1310		Non-Recur: 547 Recur: 56 **Total:** 604	$ %

Figure 3.11 Time Display for Each Main Step

This display shows the user the entire time breakdown, which is quite useful. The user can play around with the model adding and/or removing steps and immediately see the results of the changes. Non-recurring time gives the user an idea of "the time that only needs to be spent once". In this model, the non-recurring time is the setup time of each step. The recurring time depends on how many parts the user wants to make. Base on how many parts are to be made, the user can make the proper decision on how to produce them. S/he can do a sensitivity analysis on the number of parts, then see all the results on the time breakdown display.

The result of this example can be seen in Figure 3.12.



	Tool Setup	Material Setup	Layup	Debulk	Vacuum Bagging	Autoclave Setup	Cure	Finishing	User Additional
Non recurring	44	24	0	10	23	65	483	12	20
Recurring	17	34	131	94	62	51	7	31	0
Total	61	58	131	103	85	116	490	42	20

Figure 3.12 Process Time Breakdown for Each Main Step

The main steps that consume the most time are cure, layup, autoclave setup, and autoclave setup, in this respective order. These three main steps are responsible for 67% (737 min /1106 min) of total process time. User may decide to use the concept of "vital few, and trivial many" [Neoh] in the model. After the derivation of the scaling relationship between the dominating steps and the other steps, the total time can be roughly estimated by scaling up the dominating steps only.

3.9 Cost Calculation

After obtaining all the time, one can next calculate the cost. Multiplying the labor rate and machine rate to the time gives the cost. The default labor rates are given to each of the main steps. The user, as always, has the ability to change these rates. Once the user clicks "Calculate Total Cost", JavaScript uses the values, which have been derived from resources database or inputted by user, to calculate cost. The cost given is:

- Material cost from the earlier material calculation.
- Labor cost or the direct cost to be paid to the workers. It breaks down into non-recurring and recurring cost. The user can obtain the necessary information on how to best organize the workforce and how to plan the manufacturing systems. For non-recurring cost, it might use a different crew than recurring cost. Maintenance crew could come in the morning to setup for each step. Thereafter, other workers can work on recurring tasks.
- Machine cost, which is the cost of the machine(s) involved in making the part(s), can also be broken down to non-recurring and recurring costs. The same explanation, given to the labor cost, can be applied to machine cost. The maintenance could be the ones to setup the machine (non-recurring).
- Total cost is the summation of material cost, labor cost, and machine cost.
- Average cost is the cost per part. It is useful when comparing the cost between different production sizes and different processes.
- Σ(Labor cost, machine cost)shows the totaling cost of all process steps in the particular main step. This is shown in the time breakdown display in section 3.7.
- %(Total cost) for the labor and machines gives information on the relative magnitude of each main step. The user can see where and what requires the most cost. The effect of "vital few and trivial many" can be shown here as well.

In this example, the cost calculation is explained in Figure 3.13.

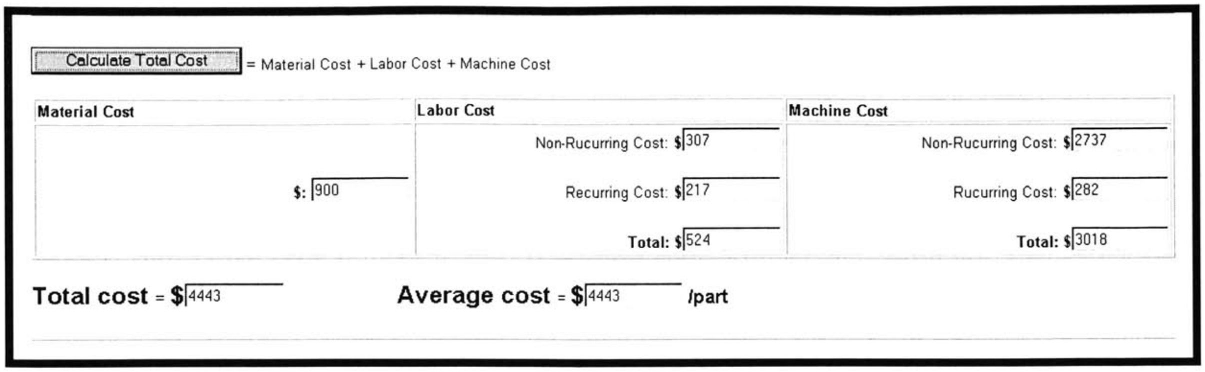


Figure 3.13 Cost Display

This concludes the example of making one simple curved part of width of 60 inches, radius of 46 inches, angle of 30 degrees, and part thickness of 0.2 inch. The material used for making this part is Fabric std. Mod-3K-70-PW, 42" untoughened. The total cost is $4443 and the average cost is $4443/part. This can be further broken down to a material cost of $900, labor cost of $524, and machine cost of $3018. If many parts are made, the average cost should go down because of the distribution of non-recurring time to parts.

Another benefit of this model is to facilitate the decision making of the user. For example, if the user is uncertain between Hand Layup and RTM in making particular parts, s/he could easily use Hand Layup and RTM models in order to make the better decision. Using the same dimension, quantity, and using the default process plan with one user additional step, the user can obtain the average cost per part in determining the more optimal process choice. Figure 3.14 summarizes the results.

Hand Layup		RTM	
# part	average cost per part	# part	average cost per part
1	4443	1	1703
10	1703	10	1080

Figure 3.14 Table summarizing the Hand Layup vs. RTM Cost

The number of part uses in determining the process is "1" and "10" although other values could be used depending on the production volume goal. In making only one part, the RTM would cost $1703/part, while the Hand Layup would be $4443/part. In making ten parts, the RTM would cost $1080/part compared to $1703/part for the Hand Layup. One needs to be very careful before concluding that the RTM is cheaper. Such a conclusion is not entirely true since several other important factors need to be taken into consideration. RTM is more capital intensive. It requires many machines, such as resin injection machine, and oven. In comparison, the Hand Layup process requires only a curing machine and vacuum compaction machine. If the user already owns the special machinery, then using RTM would be the preferred choice. If not, capital cost analysis is required before making this important decision.

The above example illustrates one way of using the CEM. Once the user has become more familiar with the model and its options, s/he will be able to use the CEM for even more benefits.

Chapter 4 Coding for the Cost Estimation Model (CEM)

This will explain the coding used throughout the CEM. Because the structure is the same between models, understanding the coding of one model should allow the programmer to apply the coding to other models. This chapter is primarily intended for the programmer in case maintenance or updating is needed.

This chapter is broken down into several parts, which are:

- Variables
- XML databases
- Dimensioning
- View of Databases
- Get values from Databases
- Calculation of material cost
- Calculation of time
- Calculation of total cost
- HTML

4.1 Variables

Variables used in all models can be categorized into these seven different groups:

1. Dimensions

Volume	v
Radius	r
Angle	delta, deldelta
Length	ly, lx1, lx2, l, arc
Thickness	t
Area	a, ba1, ba2, ta
Perimeter	p, bp1, bp2, tp

2. Material

Density	md
Rate	mr
Scrap	ms
Cost	mcost
Ply thickness	pt
Ply width	pw
Number of ply	n
Composite array	composite[ ]
Material Name	matname
Assignment of material info	comand
Material selection index	i

3. Quantity

Part made per setup (part per batch)	ppb
Number of setup (batch per shipset)	bps
Total quantity	q

4. Process

Each process contains a different process plan and thus contains different process steps. As shown in the previous chapter, each process plan contains several main steps with each main step consists of many steps. The tables, below, show only a couple of the main steps from Hand Layup. Other processes (ATP, Forming, Pultrusion, and RTM) have their own process plans and main steps. For example, Pultrusion contains ppul (pultrusion main step) and Forming contains pfc (forming cycle main step).

Step	pts, pms, …
Non recurring	ptsnon, pmsnon, …
Recurring Step array	ptsrec, pmsrec, … stp[ ]
Parameter array	par[ ]

1/0 step "checked" ?	x[ ]
Percentage array (for eqn #6)	pc[ ]
1/0 machine "used" ?	y[ ]
Number of operation per run array	num[ ]
Parameter estimator - user additional steps	z1, z2, z3, z4

5. Labor

Labor variables' names vary depending upon the process involved. For example, Pultrusion contains lpul and Forming contains lfc (forming cycle).

Step	lts, lms, ...
Non recurring	ltsnon, lmsnon, ...
Recurring	ltsrec, lmsrec, ...
Cost	lcost
Non recurring cost	lcostnon
Recurring cost	lcostrec
Labor array	lbr[ ]

6. Equipment or Machine

Equipment or machine variables' names will also vary. For example, Pultrusion contains epul and Forming contains efc (forming cycle).

Step	ets, ems, ...
Non recurring	etsnon, emsnon, ...
Recurring	etsrec, emsrec, ...
Cost	etscost, emscost, ...
Non recurring cost	etscostnon, emscostnon, ...
Recurring cost	etscostrec, emscostrec, ...
Total equipment or machine cost	ecost
Non recurring total equipment or machine cost	ecostnon
Recurring total equipment or machine cost	ecostrec

Resources step array	resstp[ ]
Resources parameter array	respar[ ]

7. Total or Summing

Sum of process time	sum1
Sum of labor time	sum2
Sum of equipment or machine time	sum3
Sum of non-recurring time	sumnon1, sumnon2, sumnon3
Sum of recurring time	sumrec1, sumrec2, sumrec3

4.2 Extensible Markup Language (XML) Databases

If the HTML is about displaying the information, XML is about describing and storing of information. A XML document is self-describing since each document contains its own set of rules to which the data must conform. In our models, we use XML for storing data while the HTML doesn't contain much content. Basically, the HTML page is only used for formatting, displaying and inputting while XML stores most of the content. The two languages, XML and HTML, are the complementary to one another.

4.2.1 Coding for XML

XML is basically a database containing text. It can be broken down to classes, base class and subclass. For our Process database, the base class is process, and the subclass is step. Each subclass contains elements (process parameters, in our Process database), which store information to be used in the calculations. Uniquely, XML has a nesting process where one object can be embedded within another. This element nesting sets up the parent/child relationship. The nesting structure is organized as:

```
<document>
 <parent1>
  <child1></child1>
  <child2></child2>
 </parent1>
</document>
```

```xml
<?xml version="1.0" ?>
<?xml-stylesheet type="text/xsl" href="Processform.xsl"?>
<Processes>

<Step>
<Description>Abrade_part_surface</Description>
<CostMethPID>10</CostMethPID>
<Equation>6</Equation>
<Setup>2</Setup>
<Delay>0.33</Delay>
<Vo1>94.39</Vo1>
<Tau1>0.96</Tau1>
<Vo2>0</Vo2>
<Tau2>0</Tau2>
<Vo3>0</Vo3>
<Tau3>0</Tau3>
<VarDesc1>interface_area</VarDesc1>
<VarDesc2>%_of_interface_area</VarDesc2>
<VarDesc3>0</VarDesc3>
<VarDesc4>qty_of_interfaces</VarDesc4>
<MinCrew>1</MinCrew>
<BaseCrew>1</BaseCrew>
<MaxCrew>4</MaxCrew>
<CycleProp>1</CycleProp>
</Step>
   .
   .
<Step>
<Description>Open Upper Frame</Description>
<CostMethPID>6080</CostMethPID>
<Equation>1</Equation>
<Setup>0</Setup>
<Delay>2</Delay>
<Vo1>0</Vo1>
<Tau1>0</Tau1>
<Vo2>0</Vo2>
<Tau2>0</Tau2>
<Vo3>0</Vo3>
<Tau3>0</Tau3>
<VarDesc1>input</VarDesc1>
<VarDesc2>0</VarDesc2>
<VarDesc3>0</VarDesc3>
<VarDesc4>0</VarDesc4>
<MinCrew>1</MinCrew>
<BaseCrew>1</BaseCrew>
<MaxCrew>4</MaxCrew>
<CycleProp>1</CycleProp>
</Step>

</Processes>
```

Declare:
version 1.0

Process Step
ID = 10
19 elements

Continue
until the last
step

Figure 4.1 Process Database

Figure 4.1 shows the uncomplicated process database. The current version of XML is 1.0, which has to be declared at the beginning of the coding. Our XML database uses the XSL style in presenting which will be discussed in the next section. The information is given between <> and </>. All steps are contained in <processes> and </processes> and all elements are contained in <step> and </step>. There is no limitation in the size of the XML database. Because all the information is stored as text, it doesn't consume much working memory of the computer.

4.2.2 Extensible Stylesheet Language (XSL)

XSL is a style sheet-based formatting language that can take XML data and produce a wide range of output results. XSL is an application of XML, so the structure and syntax of XSL are the same as those of XML. In our model, the XSL format makes it easier for the user to see the entire database. We have made the XSL in table format, so that the user can easily scroll up and down to see all the information. The coding for XSL can be seen in Figure 4.2, which is the Process database XSL.

<TR> is a symbol for row, and <TD> is a symbol for column. First the coding for header is written between <TR> and </TR>. It is broken down into 16 columns, each within <TD> and </TD>. Secondly, the values from the XML database are collected and shown as the XSL structure. The coding is {<xsl:for-each select="Processes/Step">} so that the value for "Step" is obtained within "Processes" class. Each element is collected and shown within each <td> and </td>. The coding is {<xsl:value-of select="elment name"/>}. XSL will keep collecting data, using looping, until all data is acquired. At the end of the loop, the coding {</xsl:for-each>} has to be declared. To add more column to the template, one needs to add <td></td> to both the heading and the actual data row. To ensure proper functioning, one must make sure that the heading and data row have the same number of columns.

The output of this XSL, which is linked to the XML database, is shown in Figure 4.3.

```
<?xml version="1.0" ?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/TR/WD-xsl">
<xsl:template match="/">
<html>

<head>
<title></title>
</head>

<body>
<table BORDER="2">
 <tr>
  <td><font face="Arial" size="2"><b>Description</b></font></td>
  <td><font face="Arial" size="2"><b>Process ID Number</b></font></td>
  <td><font face="Arial" size="2"><b>Equation Number</b></font></td>
  <td><font face="Arial" size="2"><b>Setup</b></font></td>
  <td><font face="Arial" size="2"><b>Delay</b></font></td>
  <td><font face="Arial" size="2"><b>Vo 1</b></font></td>
  <td><font face="Arial" size="2"><b>Tau 1</b></font></td>
  <td><font face="Arial" size="2"><b>Baseline Crew</b></font></td>
  <td><font face="Arial" size="2"><b>Vo2</b></font></td>
  <td><font face="Arial" size="2"><b>Tau2</b></font></td>
  <td><font face="Arial" size="2"><b>Vo3</b></font></td>
  <td><font face="Arial" size="2"><b>Tau3</b></font></td>
  <td><font face="Arial" size="2"><b>VarDesc1</b></font></td>
  <td><font face="Arial" size="2"><b>VarDesc2</b></font></td>
  <td><font face="Arial" size="2"><b>VarDesc3</b></font></td>
  <td><font face="Arial" size="2"><b>VarDesc4</b></font></td>

 </tr>
<xsl:for-each select="Processes/Step">
 <tr>
  <td><font face="Arial"><xsl:value-of select="Description" /></font></td>
  <td><font face="Arial"><xsl:value-of select="CostMethPID" /></font></td>
  <td><font face="Arial"><xsl:value-of select="Equation" /></font></td>
  <td><font face="Arial"><xsl:value-of select="Setup" /></font></td>
  <td><font face="Arial"><xsl:value-of select="Delay" /></font></td>
  <td><font face="Arial"><xsl:value-of select="Vo1" /></font></td>
  <td><font face="Arial"><xsl:value-of select="Tau1" /></font></td>
  <td><font face="Arial"><xsl:value-of select="BaseCrew" /></font></td>
  <td><font face="Arial"><xsl:value-of select="Vo2" /></font></td>
  <td><font face="Arial"><xsl:value-of select="Tau2" /></font></td>
  <td><font face="Arial"><xsl:value-of select="Vo3" /></font></td>
  <td><font face="Arial"><xsl:value-of select="Tau3" /></font></td>
  <td><font face="Arial"><xsl:value-of select="VarDesc1" /></font></td>
  <td><font face="Arial"><xsl:value-of select="VarDesc2" /></font></td>
  <td><font face="Arial"><xsl:value-of select="VarDesc3" /></font></td>
  <td><font face="Arial"><xsl:value-of select="VarDesc4" /></font></td>

 </tr>
</xsl:for-each>
</table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

Header Section

Actual Data Section

Looping here

Figure 4.2 XSL Coding for the Process Database

Description	Process ID Number	Equation Number	Setup	Delay	Vo 1	Tau 1	Baseline Crew	Vo2	Tau2	Vo3	Tau3	VarDesc1	VarDesc2	VarDesc3	VarDesc4
Abrade_part_surface	10	6	2	0.33	94.39	0.96	1	0	0	0	0	interface_area	%_of_interface_area	0	qty_of_interfaces
Abrade_part_surface_liquid_shim	15	6	2	0.33	94.39	0.96	1	0	0	0	0	interface_area	%_of_liquid_shim_area	0	qty_of_interfaces
Apply_clamping_force_clamps	20	2	4	1	9	0	1	0	0	0	0	interface_length	0	0	qty_of_interfaces
Apply_clamping_force_straps	30	2	20	0.13	5	0	2	0	0	0	0	interface_length	0	0	qty_of_operations/run

Figure 4.3 XSL Format of the Process Database

4.2.3 Process Database

The Process database contains all the information needed to calculate the process and labor time. This database stores the information of extensive velocity, time constant and equation to be used to perform the calculation, crew size for labor time, and much more information. The process database has already been shown in Figure 4.1. The elements within each step are:

<Description> Detailed description of the process </Description>

<CostMethPID> ID </CostMehPID>

<Equation> Equation number reference </Equation>

<Setup> Time to setup equipment and prepare to do the operation </Setup>

<Delay> Time between successive parts, after the equipment has been setup </Delay>

<Vo1> Extensive velocity 1 </Vo1>

<Tau1> Dynamic time constant of the system </Tau1>

<Vo2> Extensive velocity 2 </Vo2>

<Tau2> Extensive velocity 2 </Tau2>

<Vo3> Extensive velocity 3 </Vo3>

<Tau3> Extensive velocity 3 </Tau3>

<VarDesc1> Variable that is input to the cost equation 1 </VarDesc1>

<VarDesc2> Variable that is input to the cost equation 2 </VarDesc2>

<VarDesc3> Variable that is input to the cost equation 3 </VarDesc3>

<VarDesc4> Variable that is input to the cost equation 4 </VarDesc4>

<MinCrew> Minimum crew size where the process step equation is valid </MinCrew>

<BaseCrew> Crew size used in the equation </BaseCrew>

<MaxCrew> Maximum crew size where the process step equation is valid </MaxCrew>

<CycleProp> Identifies if larger crew size reduces time linearly </CycleProp>

Figure 4.4 Elements of the Process Database

4.2.4 Material Database

The Material database contains all the material properties and information for material cost calculation. It stores the information of material name, density, price/lb, typical scrap, thickness, and width. The material database is shown in Figure 4.5.

Material Name	Material ID	Density	Price	Scrap	Thickness	Width
Adhesive Film Grade 3	0	1	53	15	1	1
Adhesive Film Grade 5	1	1	87	15	1	1
Adhesive Foaming	2	1	14	15	1	1
Core HC 45 glass/phenolic 1/8-5.5	3	1	65.8	15	1	1
Core HC 45 glass/phenolic 1/8-8.0	4	1	71.9	15	1	1
Core HC carbon/polyimide 3/16-10.0	5	1	229	15	1	1
Core HC carbon/polyimide 3/16-6.0	6	1	268	15	1	1
Core HC carbon/polyimide 3/16-8.0	7	1	325	15	1	1
Core HC aramid/phenolic 1/8-5.0	8	1	29.6	15	1	1
Core HC aramid/phenolic 1/8-6.0	9	1	26.6	15	1	1
Core HC aramid/phenolic 1/8-8.0	10	1	21.6	15	1	1
Core HC aramid/phenolic 1/8-9.0	11	1	20.2	15	1	1
Core HC 0/90 glass/phenolic 3/16-12.0	12	1	18.22	15	1	1

⋮

Figure 4.5 Material Database

The elements within each step are:

<CompositID> ID </CompositID>

<Name> Name of material </Name>

<Density> lb/ in^3 </Density>

<Price> $/lb </Price>

<Scrap> Typical scrap rate (%) </Scrap>

<Thickness> Thickness (inch) </Thickness>

<Width> Width (inch) </Width>

<A> Note 1 </A>

<B> Note 2 </B>

Figure 4.6 Elements of the Material Database

The XSL template of the Material database has the same coding structure with XSL of the Process database. If one needs to add more information, one can add it to <A></A> or <B></B>. The XSL coding is shown in Figure 4.7.

```
<?xml version="1.0" ?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/TR/WD-xsl">
<xsl:template match="/">
<html>

<head>
<title></title>
</head>

<body>
<table BORDER="2">
 <tr>
  <td><font face="Arial" size="2"><b>Material Name</b></font></td>
  <td><font face="Arial" size="2"><b>Material ID</b></font></td>
  <td><font face="Arial" size="2"><b>Density</b></font></td>
  <td><font face="Arial" size="2"><b>Price</b></font></td>
  <td><font face="Arial" size="2"><b>Scrap</b></font></td>
  <td><font face="Arial" size="2"><b>Thickness</b></font></td>
  <td><font face="Arial" size="2"><b>Width</b></font></td>

 </tr>
<xsl:for-each select="Materials/Composite">
 <tr>
  <td><font face="Arial"><xsl:value-of select="Name" /></font></td>
  <td><font face="Arial"><xsl:value-of select="CompositID" /></font></td>
  <td><font face="Arial"><xsl:value-of select="Density" /></font></td>
  <td><font face="Arial"><xsl:value-of select="Price" /></font></td>
  <td><font face="Arial"><xsl:value-of select="Scrap" /></font></td>
  <td><font face="Arial"><xsl:value-of select="Thickness" /></font></td>
  <td><font face="Arial"><xsl:value-of select="Width" /></font></td>

 </tr>
</xsl:for-each>
</table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

Figure 4.7 XSL Coding for the Material Database

4.2.5 Resources Database

Resources Database gives the information about the machine or equipment. The default rates and cost for the program is taken from this database. These values might be a bit dated, but with regular updating, the values can be a very good representation of what is used in the industry.

The resources in the database include direct labor, autoclave equipment, Pultrusion equipment, part trim equipment, resin injection machine, oven equipment, and many more.

The coding used for this database is similar to the previous Process and Material databases. The elements for this database are shown in Figure 4.8.

```
<ResPID> ID </ ResPID >
<Resources> Name </ Resources >
<ResEqPID> Equation ID </ ResEqPID >
<Type> Capital or Labor </Price>
<K1> Resource Cost </K1>
<K2> Depreciation Schedule </K2>
<K3> $ / Cycle Hour </K3>
<K4> $ / Month </K4>
```

Figure 4.8 Elements for the Resources Database

4.2.6 JavaScript Connectivity to Databases

4.2.6.1 Loading

4.2.6.2 Viewing Database

4.2.6.3 Get Values from Material Database

4.2.6.4 Selecting Steps and Calling Process Database

4.2.6.5 Calling Resources Database

4.2.6.1 Loading

The first step in JavaScript coding is the loading of the databases. This has to be done at the beginning to make certain that the subsequent action will obtain all the information needed from databases. Please refer to the coding in Figure 4.9.

```
<body>
<script LANGUAGE="JavaScript" FOR="window" EVENT="onload">
   loadDoc()
</script>
<script
LANGUAGE="JavaScript">
      var MaterialXml = new ActiveXObject("microsoft.xmldom");
      MaterialXml.load("Materialsdh.xml");

      function loadDoc()
        {
        if (MaterialXml.readyState == "4")
          start()
        else
          window.setTimeout("loadDoc()", 4000);
        }

      function start()
        {
        var rootElem = MaterialXml.documentElement;
        }
</script>
<script LANGUAGE="JavaScript">
      var ProcessXml = new ActiveXObject("microsoft.xmldom");
      ProcessXml.load("Processdb.xml");

      function loadDoc()
        {
        if (ProcessXml.readyState == "4")
          start()
        else
          window.setTimeout("loadDoc()", 4000);
        }

      function start()
        {
        var rootElem = ProcessXml.documentElement + MaterialXml.documentElement;
        }
</script>
<script
LANGUAGE="JavaScript">
      var ResourcesXml = new ActiveXObject("microsoft.xmldom");
      ResourcesXml.load("resources.xml");

      function loadDoc()
        {
        if (ResourcesXml.readyState == "4")
          start()
        else
          window.setTimeout("loadDoc()", 4000);
        }

      function start()
        {
        var rootElem = ProcessXml.documentElement + MaterialXml.documentElement + ResourcesXml.documentElement;
        }
</script>
```

Declarartion
Where to load from ?

Loading
Start or
Timeout

Start
Active Database

Figure 4.9 Loading of the XML Databases

The coding comprises the three following parts:

1. Declaration (what and where)

    - Declare MaterialXml, ProcessXml, and ResourcesXml, in this order, to be active objects.

- Give the destination of where to load.
  - MaterialXml loads from "Materialdb.xml"
  - ProcessXml loads from "Processdb.xml"
  - ResourcesXml loads from "resources.xml"

2. Load document
   - If the XML database is ready, JavaScript calls the Start function.
   - If the database is not ready, JavaScript calls Time Out. The user has to click refresh of Internet Explorer to re-activate the databases.
3. Start
   - Make the database available to be called from JavaScript
   - Allow the rootElem variable to store the active databases.
     - First, it stores MaterialXml.documentElement
     - Secondly, the addition of ProcessXml.documentElement is stored. Thus, rootElem = ProcessXml.documentElement + MaterialXml.documentElement
     - Finally, the addition of ResourcesXml.documentElement is stored. Thus, rootElem = ProcessXml.documentElement + MaterialXml.documentElement +ResourcesXml.documentElement

4.2.6.2 Viewing Database

The user can view the databases to get further details of each database. S/he just needs to click "View Database" or "Individual Parameters". The three functions for this purpose are:

- ViewProcess
- ViewSteps
- ViewMaterials

Each function assigns the XSL or HTML page to the new window. Use the code {window.open("location","","");} to open the window. Please refer to the coding in Figure 4.10.

```
function ViewProcess(){
newWindow = window.open("Processdb.xml","","");     /*show XML TOTAL process database*/
}

function ViewSteps(){
newWindow = window.open("Process_steps.html","","");     /*show XML individual process database*/
}

function ViewMaterials(){
newWindow = window.open("Materialsdb.xml","","");     /*show XML TOTAL material database*/
}
```

Figure 4.10 Coding for Viewing of Database

4.2.6.3 Get Values from Material Database

This function, developed by Thomas Marin, an exchange student from Germany, occurs at the first step of user interface when the user selects the material. When the material is selected, JavaScript gets the index of the particular material. It then goes through the entire database, comparing the index until the correct value is found, to get all the information of the particular material. Then the default values get assigned to the JavaScript variables. The values are then displayed at the material forms. The coding is shown in Figure 4.11.

```
function getValues(form){
var composite = new createArray(10);
var comand, matName, i;
md = 0;
mr = 0;
ms = 0;

   i = form.menu.selectedIndex;      /*get the specific material from material XML database*/
   matName = "Materials/Composite[CompositID = " + i + "]";
   comand = MaterialXml.selectNodes(matName);
   composite[i] = comand;

   ms = composite[i].item(0).childNodes.item(4).text;     /*get values from selected material*/
   mr = composite[i].item(0).childNodes.item(3).text;
   md = composite[i].item(0).childNodes.item(2).text;
   pt = composite[i].item(0).childNodes.item(5).text;
   pw = composite[i].item(0).childNodes.item(6).text;

   document.material.scrap.value = ms;      /*display the values to user*/
   document.material.rate.value = mr;
   document.material.density.value = md;
   document.material.thick.value = pt;
   document.material.width.value = pw;

}
```

Declaration

Indexing, comparing, and selecting

Assigning values from elements in childnode's items to JavaScript variables

Displaying default values to HTML interface

Figure 4.11 Coding for the getValues Function

The activities involved in this function are:

- Declaring variables.
  - A composite array is created. The index, material name, and selected material information are declared. Material density, rate, and scrap are assigned to the values of 0.
- Assigning values
  - Index value is determined from the selection form.
  - Material is selected from the database.
  - All information, regarding that particular material, is stored to the array composite[i].
  - Each material has seven childnodes, starting from 0 to 6. The values from the childnodes are stored to ms, mr, md, pt, pw. For example, {md = composite[i].item(0).childNodes.item(2).text} means that the density value of the third childnode (item(2) because of starting point of 0) is passed through md. All JavaScript's material variables are assigned values here.
- Displaying the default values to HTML interface
  - Each of the form displays the default value. For example, the density is displayed to the material density form. The coding is {document.material.density.value = md;}.

4.2.6.4 Selecting Steps and Calling Process Database

This is probably the most important step for calculating the time and cost because most of the parameters used in the calculation are collected from the databases. Without getting the values, the program only provides the interface structure without any actual calculations. All equations require setup time, delay time, extensive velocity (Vol), time constant (Tau1), and base crew from the Process database. Other parameters, such as the estimated variable, number of run in the operation, and machine used, are passed to the equations directly within JavaScript.

The coding for the calling up of database and passing variables are shown in the Figure 4.12.

The important activities are:

1. Selecting of step
   - If the step is to be included, the boolean of coding {document.select.step.checked} is true. If not included, the boolean gives fault.
2. Calling the appropriate process step ID
   - The step ID passes to the XML database. After comparing the step ID to the correct childnode, the parentnode information is stored in the array. In this case, the array is named qry_i (query), with i starting from 1 and ascending. The node is selected by comparing the CostMethPID to the passing value using the coding of. {var qry_i = ProcessXml.selectNodes("Processes/Step[CostMethPID = 'Value']"); }.
3. Storing the query to array of step
   - The array stp[i–1] stores the qry_i information. Because the stp arrray starts at stp[0] and the query starts at qry1, the index in stp array is (i-1).
4. Passing of variable to be estimated
   - X[i-1] is the variable to be estimated. For equation#1, the X[i-1] is 0. For other equations, the X[i-1] can be either area, tool area, vacuum bag area, perimeter, tool perimeter, and length.
5. Giving number of run per operation
   - num[i-1] is usually one run per operation. Sometimes, num[i-1] is different than 1, as is in the debulking step below. The num[i-1] equals 0.5*number of ply, as the part is debulked every two plies.
6. Adding machine time
   - Y[i-1] determines whether that particular step uses machine or not. Y[i-1] is a boolean with value of 0 or 1. Machine time is added only when Y[i-1] = 1.
7. Calculating time using equation

- First, the equation, to be used in the calculation, is picked by the Equation ID element in the process database. It is the third element in the childnode (step). The coding are {par[2] = stp[j].item(0).childNodes.item(2).text} and {if(par[2] == 'value', then calculation time using equation #'value'}. The coding "==" is the comparison operator, while the coding "=" is the assignment operator.

- Secondly, the values of the parameters are stored. Using the same coding as picking equation#, par[3], par[4], par[5], par[6], and par[16] are setup, delay, Vol, Tau1, and base crew size, in this order.
- Finally, using the selected equation, calculate the process time, labor time, and machine time.

```
if(document.select.ts1.checked){
        var qry1 = ProcessXml.selectNodes("Processes/Step[CostMethPID = '240']");
        stp[0] = qry1;
        x[0]=ba2;                 /*estimated by ba2*/
        num[0]=1;                 /*number of operation per run of 1*/
        y[0]=0;}                  /*does not required machine in this step*/

if(document.select.db2.checked){
        var qry11 = ProcessXml.selectNodes("Processes/Step[CostMethPID = '1610']");
        stp[10] = qry11;
        x[10]=bp2;
        num[10]=n/2;              /*assume debulking every 2 plies, number of operation per run of 0.5n*/
        y[10]=0;}

if(document.select.as1.checked){
        var qry25 = ProcessXml.selectNodes("Processes/Step[CostMethPID = '2160']");
        stp[24] = qry25;
        x[24]=0;
        num[24]=1;
        y[24]=1;}                 /*required machine in this step*/


   if(par[2] == '1'){         /*equation #1 by Ein Teck Neoh*/
          par[3] = eval(stp[j].item(0).childNodes.item(3).text);       /*setup*/
          par[4] = eval(stp[j].item(0).childNodes.item(4).text);       /*delay*/
          par[16] = eval(stp[j].item(0).childNodes.item(16).text);     /*crew size*/
    sumnon1 += bps*par[3];
    sumrec1 += bps*ppb*num[j]*par[4];
    sum1 += (bps*par[3]) + bps*ppb*num[j]*par[4];
    sumnon2 += par[16]*bps*par[3];
    sumrec2 += par[16]*bps*ppb*num[j]*par[4];
    sum2 += par[16]*((bps*par[3]) + bps*ppb*num[j]*par[4]);
    sumnon3 += y[j]*bps*par[3];
    sumrec3 += y[j]*bps*ppb*num[j]*par[4];
   }
   if(par[2] == '2'){         /*equation #2 by Ein Teck Neoh*/
          par[3] = eval(stp[j].item(0).childNodes.item(3).text);       /*setup*/
          par[4] = eval(stp[j].item(0).childNodes.item(4).text);       /*delay*/
          par[5] = eval(stp[j].item(0).childNodes.item(5).text);       /*Vo1*/
          par[6] = eval(stp[j].item(0).childNodes.item(6).text);       /*Tau1*/
          par[16] = eval(stp[j].item(0).childNodes.item(16).text);     /*crew size*/
    sumnon1 += bps*par[3];
    sumrec1 += bps*ppb*num[j]*(par[4]+Math.sqrt(Math.pow(x[j]/par[5],2)+(2*par[6]*x[j]/par[5])));
    sum1 += (bps*par[3]) + bps*ppb*num[j]*(par[4]+Math.sqrt(Math.pow(x[j]/par[5],2)+(2*par[6]*x[j]/par[5])));
    sumnon2 += par[16]*bps*par[3];
    sumrec2 += par[16]*bps*ppb*num[j]*(par[4]+Math.sqrt(Math.pow(x[j]/par[5],2)+(2*par[6]*x[j]/par[5])));
    sum2 += par[16]*((bps*par[3])+ bps*ppb*num[j]*(par[4]+Math.sqrt(Math.pow(x[j]/par[5],2)+(2*par[6]*x[j]/par[5]))));
    sumnon3 += y[j]*bps*par[3];
    sumrec3 += y[j]*bps*ppb*num[j]*(par[4]+Math.sqrt(Math.pow(x[j]/par[5],2)+(2*par[6]*x[j]/par[5])));
    sum3 += y[j]*((bps*par[3])+bps*ppb*num[j]*(par[4]+Math.sqrt(Math.pow(x[j]/par[5],2)+(2*par[6]*x[j]/par[5]))));
   }
```

Selecting Step
Calling ID
Storing query to array
Passing the variables
run/operation
Machine time boolean

Calculating time

Figure 4.12 Selecting Steps and Calling Process Database

4.2.6.5 Calling Resources Database

This step occurs after all the time has been calculated and the machine time is known. The Resources database is called to obtain the machine or equipment rate ($/hr). At each main step, the machine time is added. Using that machine time within each main step, the machine or equipment rate is multiplied to obtain the machine cost. Please refer to Figure 4.13. The activities, involved in this function, are:

1. Calling the appropriate process step ID
   - The step ID passes through the XML database and is stored as the variable at $resqry_i$ (resources query), with i starting from 1 and ascending. The node is selected by comparing the ResPID to the passing value until the correct ID is found. The coding for this calling of resources is {var $resqry_i$ = ResourcesXml.selectNodes("Resources/Step[CostMethPID = 'Value']"); }.
2. Storing the query to array of resstep
   - resstp[i-1] stands for resources step. The array resstp[i–1] stores the $resqry_i$ information. Thus, the resstp arrray starts at resstp[0], while the query starts at resqry1.
3. Getting the value from the $/cycle hour element
   - JavaScript takes the value of the $/cycle hour element from the XML database. {eval} operator changes the text element to numerical element. The coding is {respar[6] = eval(resstp[j-1].item(0).childNodes.item(6).text);}, while j is the index of main step, starting at 1.
   - Then, the cost can next be calculated after converting the time from minute into hour.

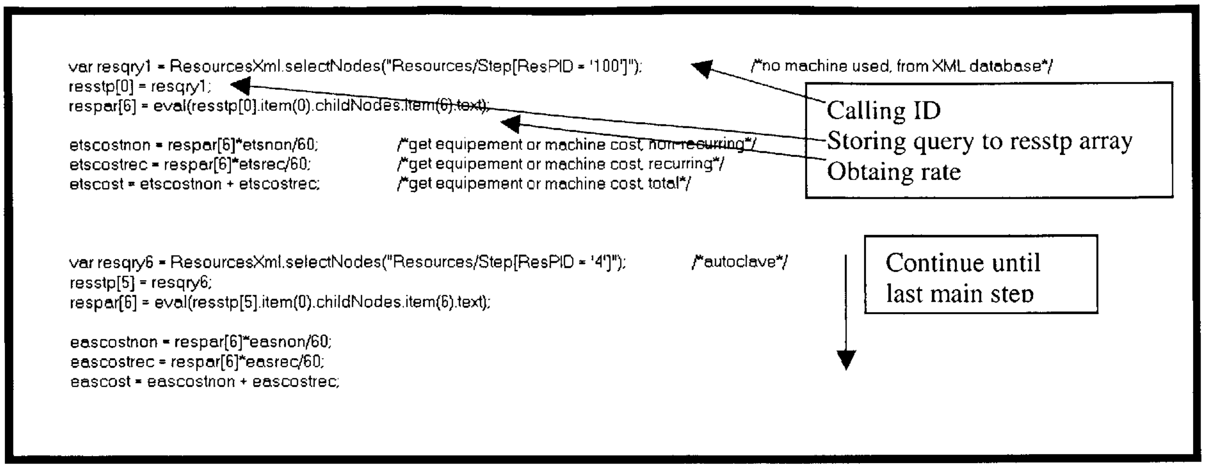


Figure 4.13 Calling of the Resources Database

4.3 Dimensioning

This section is entirely inputted by the user. After selecting the process and shape, the next step is to dimension the part. With the reference from the left side of the picture, the user inputs the dimension on the right side. The dimensioning can be categorized into two different types—length and angle whose units are inches and degrees. After the dimension has been inputted, the user has to click "Enter" to call the function Calvol. Please refer to Figure 4.14 and Figure 4.15 for the coding. Most of the variables, used in this Calvol function are the global ones. These variables are declared right at the beginning and then subsequently can be called and used throughout the entire program. However, some variables are local, such as phi in many curved parts. In the case of phi, phi only gets called and used once during this Calvol function. Phi is inputted in units of degrees, and is converted to radians in this function. After this, the angle always uses the unit of radian and Phi does not get used anymore.

Two examples are given below to show how dimensioning can be done. This same structure is applied to all other shapes and processes. This dimensioning function can be broken down into three parts.

1. Assigning values that are inputted. The value w, l, angle, h, and t are given the values from the inputted form. The coding is {var = eval(document.dimension.form.value}. The inputted value gets stored to the variable and the {eval} converts the inputted text to numerical values.

2. Calculating the relevant variables of area, volume, and many more. As seen from both Figure 4.14 and Figure 4.15, the calculation is very similar.
    - For the hand layup of flat part, the calculating variables are part area (a), ply area (ba1), vacuum bag area (ba2), part perimeter (p), ply perimeter (bp1), vacuum bag perimeter (bp2), volume (v), and number of ply (n). The area equals the part area. The assumptions used here are that ply has two inch increments from all sides from the part dimension and that the vacuum bag has two inch increments from all sides from ply dimension.
    - For Pultrusion of straight L-profile, the calculating variables are area (a), perimeter (p), volume (v), and number of ply (n). The area and perimeter in this case are cross sectional due to the characteristics of making part by Pultrusion process.
3. Displaying the result to the forms. The program gives the default number of setups as 1. Also, the calculated volume is displayed at the form.

```
function Calvol(){
  w = eval(document.dimension.width.value);              /*get values from users*/
  l = eval(document.dimension.length.value);
  t = eval(document.dimension.thickness.value);
  ms = 1 + (eval(document.material.scrap.value)/100);
  pt = eval(document.material.thick.value);
  a = w*l;
  l = 0;
  deldeltag = 0;
  Asing = 0;
  Adoub = a;
  ba1 = (w+4)*(l+4);          /*increment of 2, from each side*/
  ba2 = (w+8)*(l+8);          /*increment of 4, from each side*/
  p = 2*(l+w);
  bp1 = 2*(l+4+w+4);
  bp2 = 2*(l+8+w+8);
  v = w*l*t*ms;          /*volume calculation, including scrap*/
  n = Math.round(t/pt);          /*calculate number of plies*/
  bps = 1;
  document.quantity.batchpershipset.value = bps;          /*show default number of batch per shipset of 1*/
  document.dimension.volume.value = Math.round(v/ms);          /*show volume of part, not including scrap*/
}
```

Assigning values

Part area

Calculating variables

Displaying results

Figure 4.14 Function Calvol of the Hand Layup of Flat Part

```
function Calvol(){
  w = eval(document.dimension.width.value);      /*get values from users*/
  l = eval(document.dimension.length.value);
  h = eval(document.dimension.height.value);
  t = eval(document.dimension.thickness.value);
  ms = 1 + (eval(document.material.scrap.value)/100);
  pt = eval(document.material.thick.value);
  a = t*(h+w);              /*cross section area, going longitudinal*/
  p = 2*(h+w+t);
  v = ms*a*l;               /*volume calculation, including scrap*/
  bps = 1;
  document.quantity.batchpershipset.value = bps;           /*show default number of batch per shipset of 1*/
  document.dimension.volume.value = Math.round(v/ms);      /*show volume of part, not including scrap*/
}
```

Cross sectional area

Figure 4.15 Function Calvol of the Pultrusion Straight L-profile

4.4 Get values from material database function.

This function has already been explained in section 4.2.6.3.

4.5 Calculating material cost function

This function deals with all the information concerning the material as well as deals with the quantity to be produced. How many batches are to be produced? How many parts are in one batch? The user has the ability to change the values of the material information from the default values. This function can be broken down into four parts as folows:

1. Quantity inputted by the user. The user has to enter the number of part per setup (ppb) and number of setups (bps). The number of setups is given, by default, to 1. Multiplying this two quantity gives the total quantity.
2. Material information. The default values, from the Material XML database, for all these variables are displayed at this moment, allowing the user to overwrite new values. The material variables are material rate ($/lb), scrap (%), density (lb/in^3), ply thickness (in), and ply width (in).
3. Recalculation of volume and number of ply. Once the material information is overwritten, it changes the volume and number of ply. Thus, it needs to be updated. User needs to click "Calculate Material Cost" to do this recalculation.

4. Warnings. Because not all users will fully understand how the program works, data validation has been added. In the event of error, a warning box alerts the user and correction must be made before the user can move on to the following sections. The program makes sure that the ply thickness is smaller than the part thickness and that the number parts per setup and the number of setups are integers.

Please refer to Figure 4.16.

```
function Calmatcost(){

   ppb = eval(document.quantity.partperbatch.value);          /*user in
   bps = eval(document.quantity.batchpershipset.value);
   q = ppb*bps;          /*batch per shipset (X) part per batch gives part per shipset, same as total number of part*/
   document.material.totalquantity.value = q;
   mr = eval(document.material.rate.value);          /*user has ability to change these values, by giving values into forms*/
   ms = 1 + (eval(document.material.scrap.value)/100);
   md = eval(document.material.density.value);
   pw = eval(document.material.width.value);
   pt = eval(document.material.thick.value);
   n = Math.round(t/pt);          /*total number of plies calculation*/
   v = w*l*t*ms;
   mcost = Math.round(q*mr*md*v);          /*material calculation, in
   document.material.cost.value = mcost;          /*display material cost*/
   if(pt > t){
    window.alert("Error: Ply Thickness > Part Thickness")}
   if( (ppb%1 != 0)||(bps%1 != 0) ){
    window.alert("Error: Quantity is not an interger")}
}
```

Quantity information

Material information

Recalculations

Warnings

Figure 4.16 Calculate Material Cost Function

The coding for all the inputted quantity form is {variable = eval(document.quantity.form.value)}.and the coding for the overwrite-able material form is {variable = eval(document.material.form.value)}. This stores the inputted values. The coding for the warning is {if(condition){window.alert("Error message")}. When the user inputs incorrectly, the warning is shown as shown in Figure 4.17.

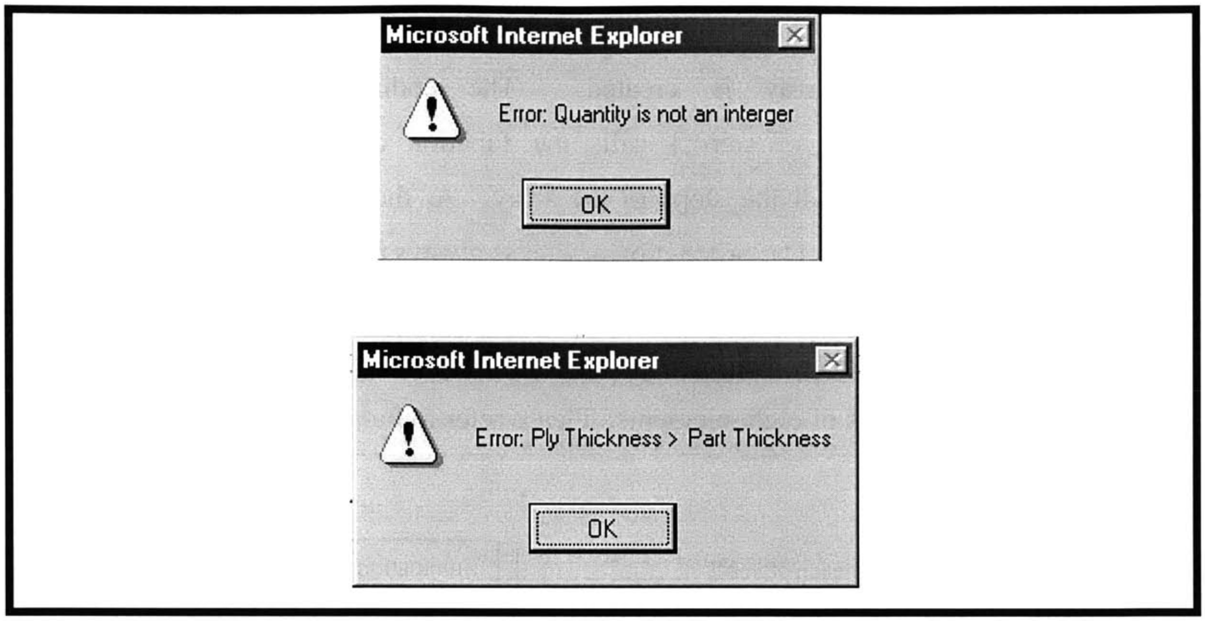


Figure 4.17 Error Statements

4.6 Calculating time function

This calculating time function section is the most important of the model. Almost all of the calculations occur here. Additionally, the longest coding also takes place here. Even though the coding is very long, the coding is repeated over and over. Nevertheless, it can be broken down into these eight parts:

4.6.1 Declare all variables and initiate values

The variables use to sum up the time are broken down to three groups. First, process time is sum1, sumnon1 (non-recurring), and sumrec1 (recurring). Secondly, labor time is sum2, sumnon2, and sumrec2. Lastly, machine time is sum3, sumnon3, and sumrec3.

Many arrays have to be declared. Arrays are used to store the information without the use of many variable names. As for the process steps, the arrays used are "stp[]" for steps used and "par[]" for parameters from the Process XML database for all process steps. The resources arrays are "resstp[]" for steps used and "respar[]" for parameters from Resources XML database.

The variables, sum1, sumnon1, sumrec1, sum2, sumnon2, sumrec2, sum3, sumnon3, and sumrec3 are initiated to 0. All the step array (stp[],par[],x[],pc[],y[], and num[]) are declared to array[number of steps]. At this declaration, the array is created. The coding {var name = new createArray(number of step);} calls the function createArray which creates memory space for all the steps in the array. At these memory spaces, "null" values are initiated. The process parameter is always declared as par[10] because it consists of 10 elements. The resources array (resstp[]) is declared to array[number of main step] and parameters array is always declared to array[8] because it consists of eight elements. Please refer to Figure 4.18.

```
function Caltime(){

var sum1, sum2, sum3;  /*summation of time*/
var sumnon1, sumnon2, sumnon3;          /*summation of nonrecurring time*/
var sumrec1, sumrec2, sumrec3;       /*summation of recurring time*/
var stp = new createArray(49);   /*array for steps*/
var par = new createArray(10);   /*array for parameters*/
var x = new createArray(49);         /*array for variable to be used in cost estimation*/
var pc = new createArray(49);    /*array for percentage( between 0 to 1) for equation#6, assume 1 for default*/
var y = new createArray(49);         /*logical, machine time calculate or not*/
var num = new createArray(49);       /*number of operation per run, usually assume single operation per run*/
var z1, z2, z3, z4;/*parameter selection for each user additional step*/
var resstp = new createArray(9);     /*array of resource steps*/
var respar = new createArray(8);     /*array of resource parameters*/

      sum1=0;              /*give initial values of 0 for these values*/
      sumnon1=0;
      sumrec1=0;
      sum2=0;
      sumnon2=0;
      sumrec2=0;
      sum3=0;
      sumnon3=0;
      sumrec3=0;
```

Summation of time variables

Process arrays

User additional steps' variables

Resources arrays

Initiating values

Figure 4.18 Declaring and Initiating the Hand Layup Process

4.6.2 Selecting steps

The section "*Selecting Steps and Calling Process Database*" has already been explained in section 4.2.6.4. Please refer to Figure 4.12 for coding.

4.6.3 Summing the time

The process, labor, and machine times are summed up in this section. The time gets calculated, using the equation needed for that particular step. The manipulative operator {+=} is used, which x += y equals to x = x+y. Using the looping coding, the selected steps keeps on being summed. The coding for this is {for(var j=0;j<number of step;j++)}. It keeps on looping, as long as the index (j) is lower than the number of steps. Then, the step selects the equation to be used. If step does not get selected, it will retain the "null" value and not be summed up. Please refer Figure 4.19.

```
for(var j=0;j<49;j++){

        if(stp[j] == null){
         sumnon1 +=0;
         sumrec1 +=0;
         sum1 +=0;
         sumnon2 +=0;
         sumrec2 +=0;
         sum2 +=0;
         sumnon3 +=0;
         sumrec3 +=0;
         sum3 +=0;}
        else{
         par[2] = stp[j].item(0).childNodes.item(2).text;

         if(par[2] == '1'){          /*equation #1 by Ein Teck Neoh*/
                par[3] = eval(stp[j].item(0).childNodes.item(3).text);         /*setup*/
                par[4] = eval(stp[j].item(0).childNodes.item(4).text);         /*delay*/
                par[16] = eval(stp[j].item(0).childNodes.item(16).text);       /*crew size*/
          sumnon1 += bps*par[3];
          sumrec1 += bps*ppb*num[j]*par[4];
          sum1 += (bps*par[3]) + bps*ppb*num[j]*par[4];
          sumnon2 += par[16]*bps*par[3];
          sumrec2 += par[16]*bps*ppb*num[j]*par[4];
          sum2 += par[16]*((bps*par[3]) + bps*ppb*num[j]*par[4]);
          sumnon3 += y[j]*bps*par[3];
          sumrec3 += y[j]*bps*ppb*num[j]*par[4];
          sum3 += y[j]*(bps*par[3]+(bps*ppb*num[j]*par[4]));
         }

                      :
```

Steps not selected do not get summed up

Steps using Equation#1 get summed up

Continue with other Equations

Figure 4.19 Summing Up the Time for Hand Layup Example

4.6.4 Storing time values to main step variables

The time values are stored to the main step variables. Then the values are passed to the display forms using the coding of {document.select.form.value = Math.round(main step value);}. The {Math.round} is a math object used to round values to integers. Please refer to coding in Figure 4.20.

```
if(j=='8'){                    /*layup*/

  plunon = sumnon1-pmsnon-ptsnon;
  plurec = sumrec1-pmsrec-ptsrec;
  plu = sum1-pms-pts;
  llunon = sumnon2-lmsnon-ltsnon;
  llurec = sumrec2-lmsrec-ltsrec;
  llu = sum2-lms-lts;
  elunon = sumnon3-emsnon-etsnon;
  elurec = sumrec3-emsrec-etsrec;
  elu = sum3-ems-ets;
  var resqry3 = ResourcesXml.selectNodes("Resources/Step[ResPID = '100']");          /*100 is the null resource values*/
    resstp[2] = resqry3;
    respar[6] = eval(resstp[2].item(0).childNodes.item(6).text);
    elucostnon = respar[6]*elunon/60;
    elucostrec = respar[6]*elurec/60;
    elucost = elucostnon + elucostrec;

 document.select.ptimenon3.value = Math.round(plunon);
 document.select.ptimerec3.value = Math.round(plurec);
 document.select.ptime3.value = Math.round(plu);
 document.select.ltimenon3.value = Math.round(llunon);
 document.select.ltimerec3.value = Math.round(llurec);
 document.select.ltime3.value = Math.round(llu);
 document.select.etimenon3.value = Math.round(elunon);
 document.select.etimerec3.value = Math.round(elurec);
 document.select.etime3.value = Math.round(elu);
 }
if(j=='10'){                   /*debulking*/
 pdbnon = sumnon1-plunon-pmsnon-ptsnon;
 pdbrec = sumrec1-plurec-pmsrec-ptsrec;
 pdb = sum1-plu-pms-pts;
 ldbnon = sumnon2-llunon-lmsnon-ltsnon;
 ldbrec = sumrec2-llurec-lmsrec-ltsrec;
 ldb = sum2-llu-lms-lts;
 edbnon = sumnon3-elunon-emsnon-etsnon;
 edbrec = sumrec3-elurec-emsrec-etsrec;
 edb = sum3-elu-ems-ets;
 var resqry4 = ResourcesXml.selectNodes("Resources/Step[ResPID = '100']");
  resstp[3] = resqry4;
  respar[6] = eval(resstp[3].item(0).childNodes.item(6).text);
  edbcostnon = respar[6]*edbnon/60;
  edbcostrec = respar[6]*edbrec/60;
  edbcost = edbcostnon + edbcostrec;

 document.select.ptimenon4.value = Math.round(pdbnon);
 document.select.ptimerec4.value = Math.round(pdbrec);
 document.select.ptime4.value = Math.round(pdb);
 document.select.ltimenon4.value = Math.round(ldbnon);
 document.select.ltimerec4.value = Math.round(ldbrec);
 document.select.ltime4.value = Math.round(ldb);
 document.select.etimenon4.value = Math.round(edbnon);
 document.select.etimerec4.value = Math.round(edbrec);
 document.select.etime4.value = Math.round(edb);
}
            :
```

Storing time values to main step' variables

Get rate
Pass to resources' array
Calculate machine cost

Storing time values to main step' variables

Figure 4.20 Storing Values to the Main Step Variables

4.6.5 Calculating machine cost

Once the time values are stored into the JavaScript variables, the machine time variables are used to obtain the machine cost. It uses the machine and equipment rates from the Resources XML database. This procedure, using resources array, is performed for all the main steps except the user additional steps. The coding, for picking the desired resources ID, is {var $resqry_i$ = ResourcesXml.selectNodes("Resources/Step[ResPID ='ID']"):}, where i is the index for the main step. After the proper resource is picked, the rate has to be assigned to that resource parameter. The coding is {respar[6] = eval(resstp[I-1].item(0).childNodes.item(6).text);}. Thereafter , the rate is multiplied to each of machine time. Refer to Figure 4.20.

4.6.6 *Adding user additional steps time*

The user additional step times give the user the opportunity to customize his/her process plan by adding up to four steps. The user has to pick the variable to be estimated using area, perimeter, and length as possible choices. The user also needs to input the values of setup time, delay time, time constant, extensive velocity, crew size as well as check whether a machine is needed. The machine rate can be entered later in the display section and will be discussed in 5.6.7. If the time constant or extensive velocity is entered as 0, equation #1 is selected but otherwise, equation #2 is automatically assumed. z1, z2, z3, and z4 are the estimated variables used in the user additional steps, in this ascending order. Please refer to the coding Figure 4.21. If an user additional step is selected, the coding {if(document.select.$additional_i$.checked)} will be true. The index (i) goes from 1 to 4. For the coding {if(document.select.avd_i.value == 'k'}, the estimating variable is area when k equals 1, perimeter when k equals 2, and length when k equals 3.

```
if(document.select.additional1.checked){          /*1st user additional step, user has input ability*/
  if (eval(document.select.avo11.value) == 0 || eval(document.select.atau1.value) == 0){
   sumnon1 += bps*eval(document.select.asu1.value);
   sumrec1 += ppb*eval(document.select.adl1.value);
   sum1 += (bps*eval(document.select.asu1.value)) + ppb*eval(document.select.adl1.value);
   sumnon2 += bps*eval(document.select.acrew1.value)*eval(document.select.asu1.value);
   sumrec2 += ppb*eval(document.select.acrew1.value)*eval(document.select.adl1.value);
   sum2 += eval(document.select.acrew1.value)*((bps*eval(document.select.asu1.value)) + ppb*eval(docu

   if(document.select.amac1.checked){          /*user choose whether the machine is used in this step
    sumnon3 += bps*eval(document.select.acrew1.value)*eval(document.select.asu1.value);
    sumrec3 += ppb*eval(document.select.acrew1.value)*eval(document.select.adl1.value);
    sum3 += eval(document.select.acrew1.value)*((bps*eval(document.select.asu1.value)) + ppb*eval(document.select.adl1.value))
                              }else{
    sumnon3 += 0;
    sumrec3 += 0;
    sum3 += 0;}
                              }
  else{
   if (document.select.avd1.value=='1'){          /*estimate by area*/
   z1=a;

   sumnon1 += bps*eval(document.select.asu1.value);
   sumrec1 += ppb*(eval(document.select.adl1.value)
           + Math.sqrt(Math.pow((z1/eval(document.select.avo11.value)),2)
           +(2*eval(document.select.atau1.value)*z1/eval(document.select.avo11.value))));
   sum1 += (bps*(eval(document.select.asu1.value)) + ppb*(eval(document.select.adl1.value)
        + Math.sqrt(Math.pow((z1/eval(document.select.avo11.value)),2)
        +(2*eval(document.select.atau1.value)*z1/eval(document.select.avo11.value)))));
   sumnon2 += bps*eval(document.select.acrew1.value)*eval(document.select.asu1.value);
   sumrec2 += eval(document.select.acrew1.value)*ppb*(eval(document.select.adl1.value)
           + Math.sqrt(Math.pow((z1/eval(document.select.avo11.value)),2)
           +(2*eval(document.select.atau1.value)*z1/eval(document.select.avo11.value))));
   sum2 += eval(document.select.acrew1.value)*((bps*eval(document.select.asu1.value)) + ppb*(eval(document.select.adl1.value)
        + Math.sqrt(Math.pow((z1/eval(document.select.avo11.value)),2)
        +(2*eval(document.select.atau1.value)*z1/eval(document.select.avo11.value)))));

    if(document.select.amac1.checked){          /*user choose whether the machine is used in this step or not*/
     sumnon3 += bps*eval(document.select.acrew1.value)*eval(document.select.asu1.value);
     sumrec3 += eval(document.select.acrew1.value)*ppb*(eval(document.select.adl1.value)
             + Math.sqrt(Math.pow((z1/eval(document.select.avo11.value)),2)
             +(2*eval(document.select.atau1.value)*z1/eval(document.select.avo11.value))));
     sum3 += eval(document.select.acrew1.value)*((bps*eval(document.select.asu1.value)) + ppb*(eval(document.select.adl1.value)
        + Math.sqrt(Math.pow((z1/eval(document.select.avo11.value)),2)
        +(2*eval(document.select.atau1.value)*z1/eval(document.select.avo11.value)))));
    }else{
     sumnon3 += 0;
     sumrec3 += 0;
     sum3 += 0;}
                              }
  if (document.select.avd1.value=='2'){          /*estimate by perimeter*/
  z1=p;
             :

 if (document.select.avd1.value =='3'){          /*estimate by length*/
 z1=arc;

             :

else{
 sumnon1 += 0;
 sumrec1 += 0;
 sum1 += 0;
 sumnon2 += 0;
 sumrec2 += 0;
 sum2 += 0;
 sumnon3 += 0;
 sumrec3 += 0;
 sum3 += 0;
}

puanon = sumnon1-pfnnon-pcunon-pasnon-pvbnon-pdbnon-plunon-pmsnon-ptsnon;
puarec = sumrec1-pfnrec-pcurec-pasrec-pvbrec-pdbrec-plurec-pmsrec-ptsrec;
pua = sum1-pfn-pcu-pas-pvb-pdb-plu-pms-pts;
luanon = sumnon2-lfnnon-lcunon-lasnon-lvbnon-ldbnon-llunon-lmsnon-ltsnon;
luarec = sumrec2-lfnrec-lcurec-lasrec-lvbrec-ldbrec-llurec-lmsrec-ltsrec;
lua = sum2-lfn-lcu-las-lvb-ldb-llu-lms-lts;
euanon = sumnon3-efnnon-ecunon-easnon-evbnon-edbnon-elunon-emsnon-etsnon;
euarec = sumrec3-efnrec-ecurec-easrec-evbrec-edbrec-elurec-emsrec-etsrec;
eua = sum3-efn-ecu-eas-evb-edb-elu-ems-ets;
var resqry9 = ResourcesXml.selectNodes("Resources/Step[ResPID = '100']");          /*null(0) as default machine rate*/
resstp[8] = resqry9;
            respar[6] = eval(resstp[8].item(0).childNodes.item(6).text);
euacostnon = respar[6]*euanon/60;
euacostrec = respar[6]*euarec/60;
euacost = euacostnon + euacostrec;
document.select.mcr9.value = respar[6];     /*display the machine rate, user able to change*/
```

Checking for selected step

Equation#1 selected
Calculations
Checking for machine used

Equation#2 is assumed
avd = 1, area is estimated
avd = 2, perimeter is estimated
avd = 3, length is estimated

If user additional step is not checked, no time is added.

Storing to user additional steps' variables

Figure 4.21 Coding for the User Additional Steps

4.6.7 *Obtaining default labor rate from database*

The default rates are given by using the direct labor rate in the Resources database. The defaults are sent to all the labor rates in the main steps. The ResPID of 1 gives the information about the direct labor. Then the labdefault, labor default variable, gets the rate from the fifth element in resources database. Please refer to coding in Figure 4.22.

```
var labdatabase = ResourcesXml.selectNodes("Resources/Step[ResPID = '1']");          /*direct labor as resource*/
var labdefault = eval(labdatabase.item(0).childNodes.item(4).text);
 for(var r=0;r<9;r++){
  lbr[r] = labdefault;
 }
 document.select.lbr1.value = lbr[0];              /*display the labor rate*/
 document.select.lbr2.value = lbr[1];
 document.select.lbr3.value = lbr[2];
 document.select.lbr4.value = lbr[3];
 document.select.lbr5.value = lbr[4];
 document.select.lbr6.value = lbr[5];
 document.select.lbr7.value = lbr[6];
 document.select.lbr8.value = lbr[7];
 document.select.lbr9.value = lbr[8];
```

Get values from Resources XML database

Passing values

Figure 4.22 Obtaining the Default Labor Rate

4.6.8 *Displaying total time*

In the last section of this calculating time function, process, labor, and machine times are displayed, categorized to non-recurring, recurring, and total times. The variables are rounded, using JavaScript math object {Math.round(value)}, to integer numbers and displayed to forms. Please refer to the coding in Figure 4.23.

```
document.select.ptimenon10.value = Math.round(sumnon1);
document.select.ptimerec10.value = Math.round(sumrec1);
document.select.ptime10.value = Math.round(sum1);
document.select.ltimenon10.value = Math.round(sumnon2);
document.select.ltimerec10.value = Math.round(sumrec2);
document.select.ltime10.value = Math.round(sum2);
document.select.etimenon10.value = Math.round(sumnon3);
document.select.etimerec10.value = Math.round(sumrec3);
document.select.etime10.value = Math.round(sum3);
```

Total time is using the index = main step + 1
Here, HL has 9 main steps

Figure 4.23 Displaying Total Time (Hand Layup Example, with Nine Main Steps)

4.7 Calculating total cost function

This function calculates cost by multiplying the labor rate by the labor time. One has the ability to input new labor rates for all main steps, which are consistency broken down into non-recurring, recurring, and total. Material, labor, and machine cost are all finally displayed here as well as the total cost and average cost per part. The average cost per part gives the user knowledge to compare between process and production quantity. One can see what yields the optimal for his/her constraints.

At this function, the summation of the labor cost and machine cost for each of the main steps is shown. If the company consists of workstations, one will have good idea of which will have the most cost. Another indicator on the cost of each of the main steps is the percentage of the total time. There is a display form at each of the main steps showing what portion of cost occurs at that main step. The concept of "Vital Few and Trivial Many" [Neoh] can be explained here. Both the summation of labor and machine times, and the percentage of the total cost are displayed at the summary table in the display section.

Coding wise, the labor cost (non-recurring and recurring) are calculated here. Summing the multiplication of labor rate and labor time after first converting hours. The user additional step asks for the input of machine rate in the case that machine is checked. euacostnon, euacostrec, and euacost are the variables of user additional step machine cost. The total cost, tcost, is also calculated. The last calculations in this program are the calculation of total main step cost (labor and machine) and calculation of percentage of total cost. The form {document.select.stepcost$_i$.value} displays the main step (i) cost. The form {document..select.percent$_i$.value} displays the percentage of total cost of a particular main step (i). The last activity of the model is the displaying of the entire total process, labor, and machine costs. Please refer to coding in Figure 4.24.

```
function Caltotcost(){

  /*user can change the labor rate for each step*/
  lcostnon = (eval(document.select.lbr1.value)*ltsnon +eval(document.select.lbr2.value)*lmsnon +eval(document.select.lbr3.value)*llunon +
      eval(document.select.lbr4.value)*ldbnon +eval(document.select.lbr5.value)*lvbnon +eval(document.select.lbr6.value)*lasnon +
      eval(document.select.lbr7.value)*lcunon +eval(document.select.lbr8.value)*lfnnon + eval(document.select.lbr9.value)*luanon)/60;
      /*get non-recurring labor cost, including user additional step*/

  lcostrec = (eval(document.select.lbr1.value)*ltsrec +eval(document.select.lbr2.value)*lmsrec +eval(document.select.lbr3.value)*llurec +
      eval(document.select.lbr4.value)*ldbrec +eval(document.select.lbr5.value)*lvbrec +eval(document.select.lbr6.value)*lasrec +
      eval(document.select.lbr7.value)*lcurec +eval(document.select.lbr8.value)*lfnrec + eval(document.select.lbr9.value)*luarec)/60;
      /*get recurring labor cost, including user additional step*/

  lcost = (eval(document.select.lbr1.value)*lts +eval(document.select.lbr2.value)*lms +eval(document.select.lbr3.value)*llu +
      eval(document.select.lbr4.value)*ldb +eval(document.select.lbr5.value)*lvb +eval(document.select.lbr6.value)*las +
      eval(document.select.lbr7.value)*lcu +eval(document.select.lbr8.value)*lfn + eval(document.select.lbr9.value)*lua)/60;
      /*get total labor cost, including user additional step*/

  euacostnon = eval(document.select.mcr9.value)*euanon/60;          /*adding the entered machine r
  ecostnon = etscostnon +emscostnon +elucostnon +edbcostnon +evbcostnon +eascostnon +ecucostnc

  euacostrec = eval(document.select.mcr9.value)*euarec/60;          /*adding the entered machine r
  ecostrec = etscostrec +emscostrec +elucostrec +edbcostrec +evbcostrec +eascostrec +ecucostrec +e

  euacost = euacostnon + euacostrec;          /*adding the two entered machine rate here*/
  ecost = etscost +emscost +elucost +edbcost +evbcost +eascost +ecucost +efncost +euacost;          /*total equipment or machine cost*/

  tcost = eval(mcost) + eval(lcost) + eval(ecost);          /*total cost*/
  avecost = tcost/q;          /*average cost per part*/

  document.select.stepcost1.value = Math.round((lts*eval(document.select.lbr1.value)/60)+etscost);          /*calculate cost(labor&machine) fc
  document.select.percent1.value = Math.round(100*((lts*eval(document.select.lbr1.value)/60)+etscost)/tcost);          /*percent of total cost*/

  document.select.stepcost2.value = Math.round((lms*eval(document.select.lbr2.value)/60)+emscost);
  document.select.percent2.value = Math.round(100*((lms*eval(document.select.lbr2.value)/60)+emscost)/tcost);

  document.select.stepcost3.value = Math.round((llu*eval(document.select.lbr3.value)/60)+elucost);
  document.select.percent3.value = Math.round(100*((llu*eval(document.select.lbr3.value)/60)+elucost)/tcost);

  document.select.stepcost4.value = Math.round((ldb*eval(document.select.lbr4.value)/60)+edbcost);
  document.select.percent4.value = Math.round(100*((ldb*eval(document.select.lbr4.value)/60)+edbcost)/tcost);

  document.select.stepcost5.value = Math.round((lvb*eval(document.select.lbr5.value)/60)+evbcost);
  document.select.percent5.value = Math.round(100*((lvb*eval(document.select.lbr5.value)/60)+evbcost)/tcost);

  document.select.stepcost6.value = Math.round((las*eval(document.select.lbr6.value)/60)+eascost);
  document.select.percent6.value = Math.round(100*((las*eval(document.select.lbr6.value)/60)+eascost)/tcost);

  document.select.stepcost7.value = Math.round((lcu*eval(document.select.lbr7.value)/60)+ecucost);
  document.select.percent7.value = Math.round(100*((lcu*eval(document.select.lbr7.value)/60)+ecucost)/tcost);

  document.select.stepcost8.value = Math.round((lfn*eval(document.select.lbr8.value)/60)+efncost);
  document.select.percent8.value = Math.round(100*((lfn*eval(document.select.lbr8.value)/60)+efncost)/tcost);

  document.select.stepcost9.value = Math.round((lua*eval(document.select.lbr9.value)/60)+euacost);
  document.select.percent9.value = Math.round(100*((lua*eval(document.select.lbr9.value)/60)+euacost)/tcost);

  document.select.stepcost10.value = Math.round(lcost+ecost);          /*total process, labor, equipment or machine cost*/
  document.select.percent10.value = Math.round(100*(lcost+ecost)/tcost);          /*percent of total time, less than 100 because of material cost*/

  document.cost.mtlcost.value = Math.round(eval(mcost));          /*summary display*/
  document.cost.lbrcostnon.value = Math.round(lcostnon);
  document.cost.lbrcostrec.value = Math.round(lcostrec);
  document.cost.lbrcosttotal.value = Math.round(lcostnon +lcostrec);
  document.cost.maccostnon.value = Math.round(ecostnon);
  document.cost.maccostrec.value = Math.round(ecostrec);
  document.cost.maccosttotal.value = Math.round(ecostnon +ecostrec);
  document.cost.totcost.value = Math.round(eval(tcost));
  document.cost.avecost.value = Math.round(eval(avecost));
```

Calculating labor cost by multiplying rate with time

User additional steps
Total time calculation
Average time calculation

Stepcost
%(total time)

Final display

Figure 4.24 Calculation of the Total Cost Function

4.8 HyperText Marked-up Language (HTML)

This section explains the user interface. What user sees and uses are all in the HTML. Because of the condition of the funding from the National Science Foundation (NSF), our research group must make the model available to the public domain. Thus, the HTML format is the best solution since any user could just log-in to the model site and then conveniently use the model for his/her own purposes. Our main objective has been to make this HTML as user-friendly as possible whereas the aesthetics of the HTML has not been a priority. Still it is possible that these models might be improved aesthetically at the end of the project when the models are finished and published for the public domain. The JavaScript works with HTML four ways. First, the values used in the calculations are collected from the HTML forms. Forms, which are the most important feature in our WEB based models, have the creation of interactive HTML pages as their objective. The values of input forms are either inputted by the user or passed from the XML databases. Secondly, with the linking of pages, the user clicks on the links to transfer to the desired destinations. Thirdly, the HTML selections are passed to JavaScript. Finally, the HTML buttons call up the JavaScript functions, which in turn cause all the actions. These functions are the main actors in these models; without these functions, no calculation can be performed.

Figure 4.25 is used to summarize the key HTML players that interrelate with JavaScript.

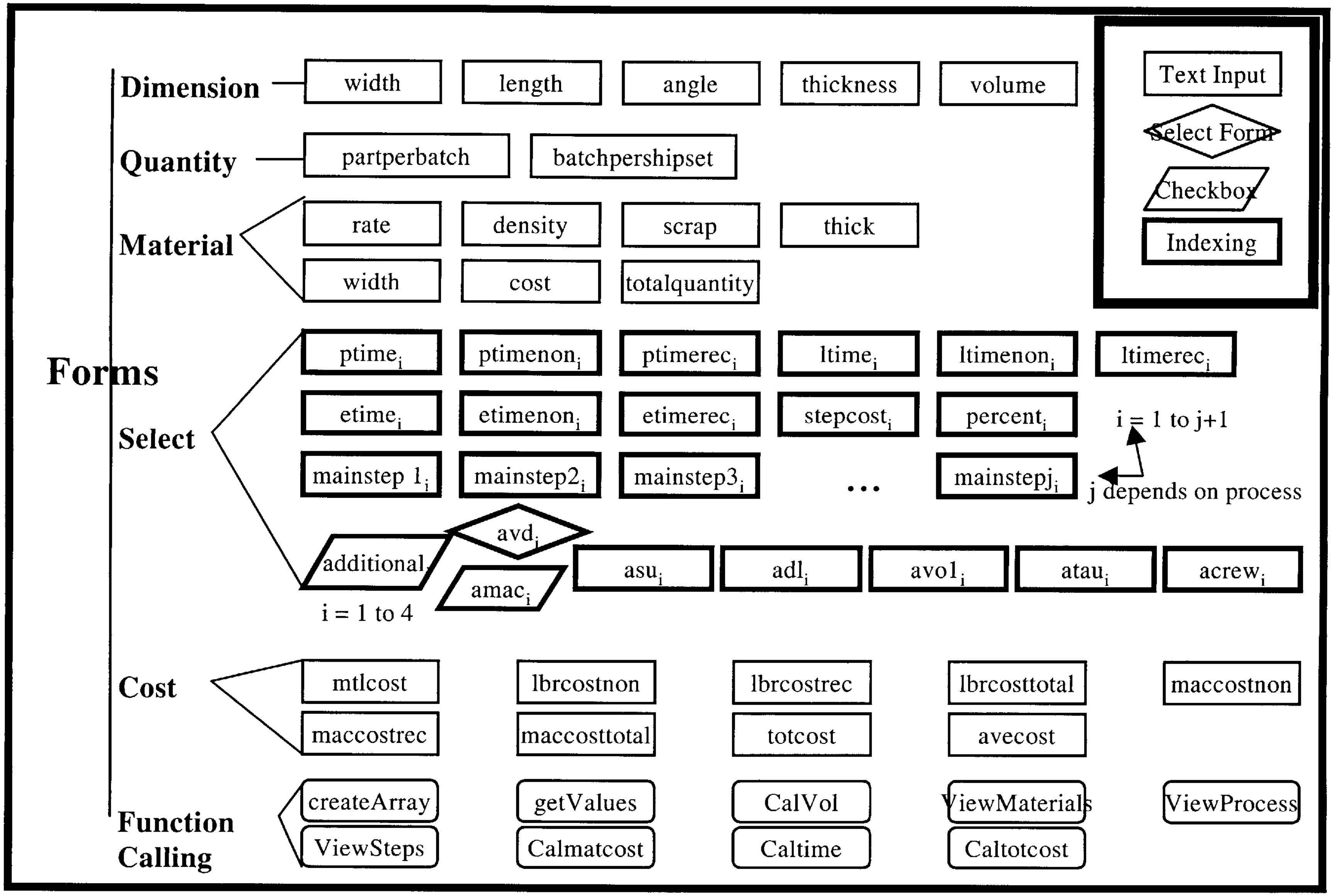


Figure 4.25 The HTML Forms

As seen in Figure 4.25, the HTML forms can be broken into these six parts:

- Dimension form. Depending on the shape, the text input names are different. With curved parts, the text inputs include angle and radius. Otherwise, the text inputs consist of width and length, which may have different names, depending on the particular shapes.
- Quantity form. The quantity information gets inputted here, using text input names are partperbatch (part made per setup) and batchpershipset (number of setup).
- Material form. All the material inputs are entered here. The text input's names are rate, density, scrap, thick, width, cost, and totalquantity.
- Select form. The majority of the HTML inputs are here. The text input's names are ptime, ptimenon, ptimerec, ltime, ltimenon, ltimerec, etime, etimenon, etimerec, stepcost, and percent. These are consistent throughout every shape and process. For the rest, the text inputs depend on what process is selected. The names are in the structure of mainstep1i, mainstep2i, ..., and mainstepji, where the j depends on the process selected and i equals j+1, because of the inputs storing the total time. User additional steps are selected at 4 checkbox inputs (additional1, ..., additional4). The checkbox object denotes the logical (true or false) data. The selection object is one of the most useful and flexible of all form objects and can takes up three different forms of selection list, scrolling list, and multiselection scrolling list. Each user additional step contains one selection input (avdi, additional step variable description), one checkbox input (amaci, additional step machine used), and five text inputs, which are asui (additional step setup), adli (additional step delay), avo1i (additional extensive velocity), ataui (additional step time constant), and acrewi (additional step crew). The index, i, at the end of all user additional step inputs goes from 1 to 4.
- Cost form. These forms, which display all the cost values at the end of the model, are all text inputs, with names of mtlcost (material cost), lbrcostnon (non-recurring labor cost), lbrcostrec (recurring labor cost), lbrcosttotal (total labor cost), maccostnon (non-recurring machine cost), maccostrec (recurring machine cost), maccosttotal (total machine cost), totcost (total cost), and avecost (average cost per part).
- Calling function buttons. These buttons are the ones that call for all the actions. They are named as "Enter", "View XXX", and "Calculate XXX". The functions that

they call forth are getValues, Calvol, VeiwMaterials, ViewProcess, ViewSteps, Calmatcost, Caltime, Caltotcost. All of these functions have already been explained earlier in this chapter.

The above are the entire HTML forms and inputs which have been divided up to make them similar to sections. Each section (form) has its own purpose, such as the material form gets material information. The input resides within the particular form. One can see that when the JavaScript variable stores value to the input, the input will be named after the form (document.form.input.value). The input breaks down to three types, text, selection, and checkbox, in our models. The text input stores text, which includes numbers. Numbers is displayed and inputted here. The selection input lets the user select the desired item. The checkbox has only two values—true and false. If "checked", it is true. All of the input names are supposedly self-explanatory. If the models need adjustment, one should be able to easily understand what is going on in the HTML coding.

Chapter 5 Process Plans

This chapter discusses the remaining four process plans (ATP, Forming, Pultrusion, and RTM) since the Hand Layup has been discussed in the Chapter 3. Base on the knowledge of the processes, the process plans were constructed. Each process consists of steps in the given sequences. [Neoh] process steps are used, if available. If not, the physical models are created to come up with the parameters.

The differences between one process plan to another are the variables declaration, Caltime function, Caltotcost function, and HTML interface. All these differences are discussed in this chapter for the purpose of any updating or improving the CEMs.

Automated Tow Placement (ATP)

The main steps of the ATP process, in order, are:

- Part and tool setup
- Machine setup
- Material setup
- Layup
- Part unloading and handling
- Part protection
- User additional steps

The process plan of the ATP process is described in Figure 5.1.

Main Step	Process Step	ID	Equation	Estimated Variable	Machine (Y/N)
Part & Tool Setup	Identify required items for TPM	550	1	0	N
	Position winding tool into staging area	1030	1	0	N
	Apply separation film to winding tool surface	885	6	Separation film area	N
	Hand lay up fabric ply over winding tool	860	2	Fabric area	N
	Position skin debulk bag	1060	2	Bag area	N
	Debulk hand layed up fabric ply	340	1	0	N
	Remove debulk bag from skin	1610	2	Bag perimeter	N
	Load winding tool onto TPM	1030	1	0	N
Machine Setup	Setup equipment for skin layup	1940	1	0	Y
Material Setup	Load prepreg tow onto ATP equipment as required	920	2	Prepreg weight	Y
Layup	Layup 0 degree plys onto winding tool using TPM	700	8	Part length and width	Y
	Layup 15 degree plys onto winding tool using TPM	710	8	Part length and width	Y
	Layup 30 degree plys onto winding tool using TPM	720	8	Part length and width	Y
	Layup 45 degree plys onto winding tool using TPM	730	8	Part length and width	Y
	Layup 60 degree plys onto winding tool using TPM	740	8	Part length and width	Y
	Layup 75 degree plys onto winding tool using TPM	750	8	Part length and width	Y
	Layup 90 degree plys onto winding tool using TPM	760	8	Part length and width	Y
Part Unloading & Handling	Unload winding tool with skin layup from TPM cell	1490	1	0	Y
Part Protection	Hand lay up fabric ply over winding tool	860	2	Fabric area	N
	Position skin debulk bag	1060	2	Bag area	N
	Debulk hand layed up fabric ply	340	1	0	N
	Remove debulk bag from skin	1610	2	Bag perimeter	N
	Protect skin layup on winding tool	1440	2	Part area	N
	Identify skin layup on winding tool	540	1	0	N
User Additional Steps	User additional step 1	none	1 or 2	Part area, perimeter, or length	Y or N
	User additional step 2	none	1 or 2	Part area, perimeter, or length	Y or N
	User additional step 3	none	1 or 2	Part area, perimeter, or length	Y or N
	User additional step 4	none	1 or 2	Part area, perimeter, or length	Y or N

Figure 5.1 Process Plan for the ATP Process [Haffner, 6]

Equations

Equations used in the ATP process are #1, #2, and #8. Equation #8 is a special equation used in the laying up step in the ATP process. This equation, which includes cross edge delay, was developed by [Beresheim]. For the simple curved part, straight L-profile, and straight tube profile, the number of cross edges is zero, one, and zero, in this order. See Figure 5.2. The equation is:

$$Time = \frac{\sum_{i=1}^{N}(normal\ placement\ time + cross\ edge\ delay + start\ /\ end\ path\ delay)}{(1 - downtime)}$$

Note: 1) N = number of plies and 2) No machine downtimes due to maintenance or machine failures are considered.

The ATP CEM follows the suggested sequence by [Beresheim]. The basic variables are declared locally at the beginning of the Caltime function where the declared variables are the safety margin (S), the ratio of interrupted to steady state velocities (rv), and the time delay through acceleration and deceleration (tap). Other basic parameters, which are setup, delay, Vo, a, and crew size, are collected from the Process database as par[3], par[4], par[5], par[6], and par[16], in this order. All other variables are calculated in the following sequence:

Dimensioning the ply orientation inputs

Pass the estimated variables, q1[] and q2[], to the Equation #8 where the q1[] = length.sin(fiber angle) and the q2[] = width.cos(fiber angle)

Calculate the number of strips (ns)

Calculate the out-of-plane angle (delta_n)

Calculate the length to cross an edge (lce)

Calculate the total length to cross an edge with lower speed (lslow)

Calculate the time delay through crossed edges (tce)

Declare the number of cross edges (nce)

Calculate the normal placement time (tpl), the crossed edge delay (ted), and the start/end delay (tse)

Sum up the three calculated times to obtain the process time

Multiply by the number of plys to obtain the total process time for the layup step

Resources

The automated tow placement machine (Resources ID = 5) is in operation during some process steps in the machine setup, material setup, and layup main steps. The trim equipment (Resources ID = 17) is utilized during some process steps in the part unloading and handling main steps.

Coding Differences

The differences in the JavaScript coding are:

1. Equation #8 has its own variables which are s, rv, tap, dt, ns, deltan, ta, lce, lslow, tce, nce, tpl, and ted which are all declared within Caltime function as local variables.
2. The step arrays are all declared to store 24 memory spaces. stp[], x[], pc[], y[], num[] are all created as {var name = new createArray(24);} and the resources array stores 6 memory spaces. The parameter arrays are the same with the other processes since the same databases (Process and Resources) are used.
3. The CheckOrientation function is added. Because the orientations of ply (0, 15, 30, 45, 60, 75, and 90 degrees) need to be inputted and used throughout the model, the variables of n0, n15, n30, n45, n60, n75, and n90 must be declared globally at the beginning of the JavaScript coding. The html interface for this section is shown in Figure 5.3 and Figure 5.4. The CheckOrientation function is explained in Figure 5.5.

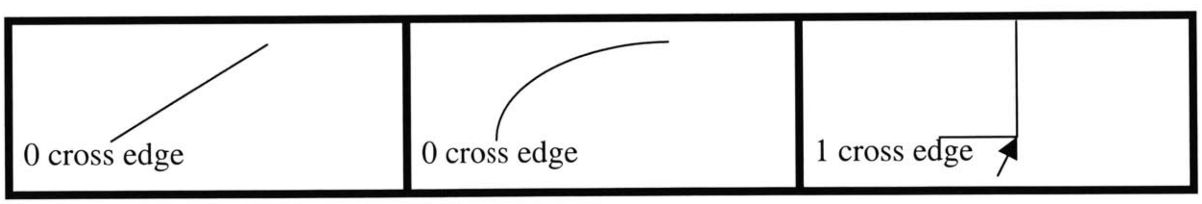


Figure 5.2 Cross Edge for the Three Selected Parts

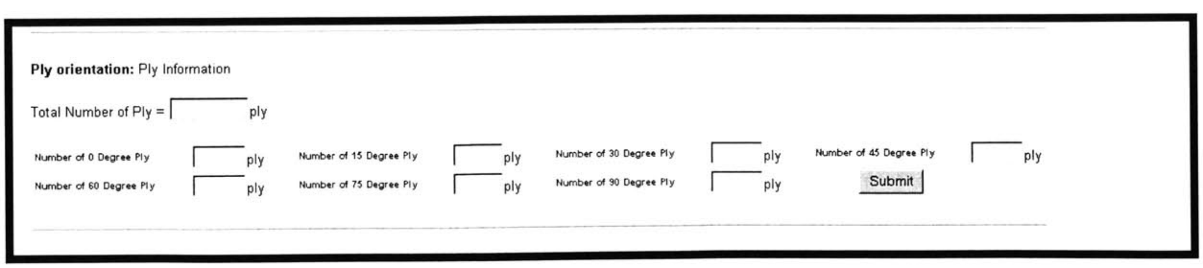


Figure 5.3 Ply Orientation Interface

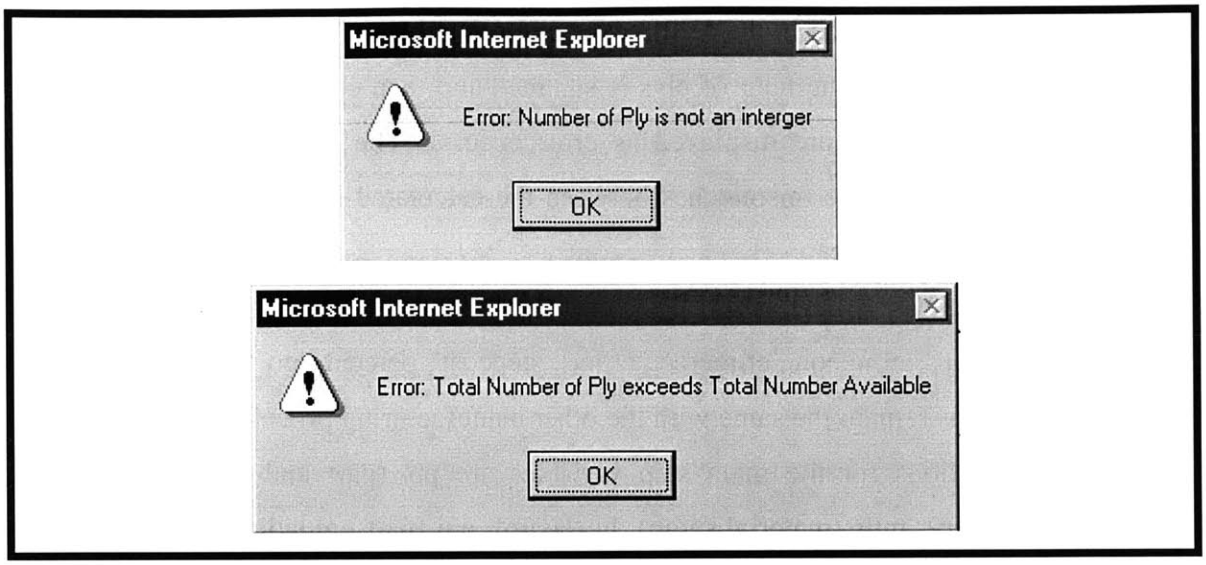


Figure 5.4 Warnings for Ply Orientation Inputs

```
function CheckOrientation(){
var nuse;
   n0 = eval(document.material.n0.value);
   n15 = eval(document.material.n15.value);
   n30 = eval(document.material.n30.value);
   n45 = eval(document.material.n45.value);
   n60 = eval(document.material.n60.value);
   n75 = eval(document.material.n75.value);
   n90 = eval(document.material.n90.value);
   nuse = n0 +n15 +n30 +n45 +n60 +n75 +n90;        /*check on the number user inputted*/
   if(nuse > Math.ceil(navailable)){
    window.alert("Error: Total Number of Ply exceeds Total Number Available")}
   if(nuse < Math.ceil(navailable)){
    window.alert("Error: Total Number of Ply not equals Total Number Available")}
   if( (n0%1 != 0)||(n15%1 != 0)||(n30%1 != 0)||(n45%1 != 0)||(n60%1 != 0)||
                    (n75%1 != 0)||(n90%1 != 0) ){
    window.alert("Error: Number of Ply is not an interger")}
}
```

Variables inputted

Error statements

Figure 5.5 JavaScript Coding for the CheckOrientation Function

Although the JavaScript coding gives default values for the ply orientation, the user still has the ability to overwrite them. Once the orientation information is inputted, the user needs to click “Submit” to call the CheckOrientation function.

As seen in Figure5.5, the orientation variables (n0, ..., n90) store the inputted values. These variables are used in the Caltime function to calculate the layup time for each orientation. The total number of plys is summed and then checked for errors. Error windows, Figure 5.4, are displayed as error occurs. The errors include the non-integer of plies and the mismatches between the calculated number of plies and the inputted number of plies.

4. HTML coding has 7 main steps. The forms, $ptimenon_i$, $ptimerec_i$, $ptime_i$, $ltimenon_i$, $ltimerec_i$, $ltime_i$, $etimenon_i$, $etimerec_i$, $etime_i$, $stepcost_i$, $percent_i$, go from 1 to 8 while the other forms remain the same with the other manufacturing processes.
5. The abbreviations for the main step variables are pts (part and tool setup), mcs (machine setup), mtls (material setup), lu (layup), puj (part unloading and handling), pp (part protection, and ua (user additional steps).

Forming

The main steps of the Forming process, in order, are:

- Material and tool setup
- Layup
- Machine setup
- Forming cycle
- Autoclave setup
- Cure
- Finishing
- User additional steps

The process plan of the Forming process is described in Figure 5.6.

Main Step	Process Step	ID	Equation	Estimated Variable	Machine (Y/N)
Material and Tool Setup	Cut material	2280	2	Trim length	N
	Clean tools	240	2	Tool area	N
	Tool setup	2160	1	0	N
	Apply release agent	50	2	Tool area	N
Layup	layup	5000	10	Part area	N
Machine Setup	Setup Forming machine	6000	1	0	Y
	Load & setup tooling	6010	1	0	Y
	Load parts	1210	2	Part length	Y
	Lower upper diaphragm frame	6030	1	0	Y
	Lock frame	6040	1	0	Y
Forming Cycle	Preheat charges	6050	5	0	Y
	Form & apply vacuum	6060	1	0	Y
	Cool parts	6070	5	Input	Y
	Open upper frame	6080	1	0	Y
Autoclave Setup	Transfer to autoclave	2160	1	0	Y
	Connect to vacuum line	150	2	Part perimeter	Y
	Connect to thermocouples				
		130	2	Part perimeter	Y
		1270	2	Part perimeter	Y
	Apply vacuum	80	1	0	Y
	Check seals	4010	5	Bag perimeter	Y
	Setup autoclave				
		300	1	0	Y
		940	1	0	Y
		2050	1	0	Y
Cure	Start autoclave cycle	350	1	0	Y
	Disconnect vacuum	1560	2	Bag perimeter	Y
	Disconnect thermocouples	1540	2	Part perimeter	Y
	Remove part from autoclave				
		4020	1	0	Y
		4030	2	Bag area	N
		4040	5	Bag perimeter	N

Main Step	Process Step	ID	Equation	Esimated Variable	Machine (Y/N)
Finishing	Demold part				
		1740	2	Peel ply area	N
		1800	6	Interface area	N
	Clean part	180	2	Interface area	N
	Abrade part	10	6	Interface area	N
	Trim part	2280	2	Trim length	N
	Deflash *	2350	2	Trim length	N
	Debur *	2340	2	Trim length	N
User Additional Steps	User additional step 1	none	1 or 2	Part area, perimeter, or length	Y or N
	User additional step 2	none	1 or 2	Part area, perimeter, or length	Y or N
	User additional step 3	none	1 or 2	Part area, perimeter, or length	Y or N
	User additional step 4	none	1 or 2	Part area, perimeter, or length	Y or N

Figure 5.6 Process Plan for the Forming Process

Equations

Equations used in the Forming process are #1, #2, #5, #6, and #10. The Equation #10 is used because the Hand Layup process is a step included in the Forming process. The steps "Preheat charges" and "Cool parts" use Equation #5 in calculating time. Haffner developed these two steps using a physical model.

Resources

The Diaphragm Forming machine (Resources ID = 11) is in operation during some steps in both the machine setup and the forming cycle main steps. In addition, the autoclave (Resources ID = 4) is used during some steps in the autoclave setup and cure main steps.

Coding Differences

1. The step arrays are all declared to store 36 memory spaces. stp[], x[], pc[], y[], num[] are all created as {var name = new createArray(36);} and the resources array stores 8 memory spaces. The parameter arrays remain the same with the other manufacturing processes.
2. HTML coding has 8 main steps. The forms, $ptimenon_i$, $ptimerec_i$, $ptime_i$, $ltimenon_i$, $ltimerec_i$, $ltime_i$, $etimenon_i$, $etimerec_i$, $etime_i$, $stepcost_i$, $percent_i$, go from 1 to 9 while the other forms remain the same.
3. The abbreviations for the main step variables are mtls (material and tool setup), lu (layup), mcs (machine setup), fc (forming cycle), as (autoclave setup), cu (cure), fn (finishing), and ua (user additional steps).

Pultrusion

The main steps of the Pultrusion process, in order, are:

- Machine setup
- Material setup
- Setup resin bath
- Setup resin injection machine
- Pultrusion
- Finishing
- Inspection
- User additional steps

The process plan of the Pultrusion process is described in Figure 5.7.

Equations

The equations used in the Forming process are #1, #2, and #7. The Equation #7 used in the Pultrusion step in the Pultrusion process was developed by [Haffner, 4] using a physical model of the Pultrusion process. Equation #7 is described as:

$$\Pr ocess\ \ time = \#\ setup \times [setup + (\#\ part\ per\ setup \times \#\ run \times (delay\ /\ v))]$$

Note: The estimated variable (v) is the thickness.

Resources

The Pultrusion machine (Resources ID = 9) is used during the main steps of the machine setup, material setup, Pultrusion. Part NDI equipment (Resources ID = 16) is in operation in some steps of the inspection main step. Part trim equipment (Resources ID = 17) is used in the finishing main step. The resin injection machine (Resources ID = 30) is used during the setup resin injection machine main step.

Main Step	Process Step	ID	Equation	Estimated Variable	Machine(Y/N)
Machine Setup	Identify required items for pultrusion	550	1	0	N
	Setup pultrusion equipment	2070	1	0	Y
	Setup inline ultra-sonic inspection equipment	2020	1	0	Y
	Load form die onto pultrusion equipment	1140	1	0	Y
	Attach thermocouple lines to pultrusion die	120	1	0	Y
Material Setup	Load dry fiber material onto pultrusion equipment	900	1	0	Y
Setup Resin Bath	Setup resin bath				
	Load resin into resin bath				
Setup Resin Injection Machine	Setup pultrusion resin injection machine	2080	1	0	Y
	Load two part resin onto pultrusion injection machine	930	2	Resin weight	Y
	Attach resin injection lines to pultrusion die	100	1	0	Y
Pultrusion	Run pultrusion equipment (including cut-off)	3000	7		Y
	Remove form die from pultrusion equipment	1660	1	0	Y
	Clean pultrusion form die	210	2	Section trace	Y
Finishing	Setup NC trimming equipment	2040	1	0	Y
	Position part into NC trimming equipment	1210	2	Part length	Y
	Trim automated edge gr/ep	2220	2	Trim length	Y
	Remove finished part from NC trimming equipment	1730	2	Part length	N
	Manually deburr edge	2340	2	Trim length	N
Inspection	Setup offline NC ultra-sonic equipment *	2030	1	0	Y
	Position part into NC ultra-sonic inspection equipment *	1000	2	Part length	Y
	Ultra-sonic inspect part in NC ultra-sonic inspection equipment *	570	2	Part length	Y
	Remove finished part from NC ultra-sonic inspection equipment *	1730	2	Part length	Y
User Additional Steps	User additional step 1	none	1 or 2	Part area, perimeter, or length	Y or N
	User additional step 2	none	1 or 2	Part area, perimeter, or length	Y or N
	User additional step 3	none	1 or 2	Part area, perimeter, or length	Y or N
	User additional step 4	none	1 or 2	Part area, perimeter, or length	Y or N

Figure 5.7 Process Plan for the Pultrusion Process [Haffner, 4]

Coding Differences

1. The step arrays are all declared to store 21 memory spaces. stp[], x[], pc[], y[], num[] are all created as {var name = new createArray(21);} and the resources array is declared to acquire 7 memory spaces. The parameter arrays are the same with other processes, since the same databases are used.
2. HTML coding has 8 main steps. The forms, $ptimenon_i$, $ptimerec_i$, $ptime_i$, $ltimenon_i$, $ltimerec_i$, $ltime_i$, $etimenon_i$, $etimerec_i$, $etime_i$, $stepcost_i$, $percent_i$ go from 1 to 9.
3. The abbreviations for the main step variables are mcs (machine setup), mls (material setup), srb (setup resin bath), srim (setup resin injection machine), pul (Pultrusion), fn (finishing), in (inspection), and ua (user additional steps).

Resin Transfer Molding (RTM)

The main steps of the RTM process, in order, are:

- Tool setup
- Material loading
- Close mold
- Setup resin injection machine
- Injection/cure
- Demolding
- Post cure
- Finishing
- Inspection
- User additional steps

The process plan of the RTM process is described in Figure 5.8.

Main Step	Process Step	ID	Equation	Estimated Variable	Machine(Y/N)
Tool Setup	Clean RTM Mold	290	2	Tool area	N
	Apply Mold Release to RTM Mold	50	2	Tool area	N
Material Loading	Position Braided Preform into Mold	960	2	Tool length	N
	Fold Frame Flange	520	2	Preform length	N
Close Mold	Position "O" Ring Mold Seal	1260	2	Seal length	N
	Close Off RTM Mold	1250	2	Tool lid perimeter	N
	Attach Vacuum Lines to RTM Mold	140	2	Injection mold length	Y
	Attach Thermocouple Lines to RTM Mold	110	2	Injection mold length	Y
	Attach Resin Injection Lines to RTM Mold	90	2	Injection mold length	Y
Setup Resin Injection Machine	Setup Resin Injection Machine	2080	1	0	Y
	Load Two Part Resin onto Pultrusion Injection Machine	930	2	Resin weight	Y
Injection/Cure	Pull Vacuum on RTM Mold	80	1	0	Y
	Cure Frame Blank in RTM Mold	380	1	0	Y
Demolding	Remove Injection Lines to RTM Mold	1500	2	Injection mold length	Y
	Remove Thermocouple lines to RTM Mold	1520	2	Injection mold length	Y
	Remove Vacuum Lines to RTM Mold	1550	2	Injection mold length	Y
	Remove RTM Lid	1770	2	Tool lid perimeter	N
	Remove RTM Seal	1780	2	Seal length	N
	Remove Frame Blank from RTM Mold	1460	2	Tool length	N
Post Cure	Post Cure Frame Blank	390	1	0	N
Finishing	Trim Manual Edge gr/ep	2280	2	Trim length	N
	Setup NC Trimming Equipment	2040	1	0	Y
	Position Part into NC Trimming Equipment	1210	2	Part length	Y
	Trim Automated Edge gr/ep	2220	2	Trim length	Y
	Remove Finished Part from NC Trimming Equipment	1730	2	Part length	Y
	Deflash Cured Frame Blank	2350	2	Trim length	N
	Manually Deburr Edge	2340	2	Trim length	N
Inspection	Setup Offline NC Ultrasonic Equipment	2030	1	0	Y
	Position Part into NC Ultrasonic Inspection Equipment	1000	2	Part length	Y
	Ultrasonic Inspect Part in NC Ultrasonic Inspection Equipment	580	2	Part area	Y
	Remove Finished Part from NC Ultrasonic Inspection Equipment	1480	2	Part length	Y
User Additional Steps	User additional step 1	none	1 or 2	Part area, perimeter, or length	Y or N
	User additional step 2	none	1 or 2	Part area, perimeter, or length	Y or N
	User additional step 3	none	1 or 2	Part area, perimeter, or length	Y or N
	User additional step 4	none	1 or 2	Part area, perimeter, or length	Y or N

Figure 5.8 Process Plan for the RTM Process [Haffner, 7]

Equations

The equations used in the Forming process are only #1 and #2. No equation has to be developed for the RTM process due to the very good documentation on this topic by [Neoh].

Resources

The RTM mold oven (Resources ID = 28) is used during the close mold, demolding, and post-cure main steps. Resin injection machine (Resources ID = 30) is in operation during the setup resin injection machine main step. The part trim equipment (Resources ID = 17) is used during the finishing main step.

Coding Differences

1. The step arrays are all declared to store 31 memory spaces. stp[], x[], pc[], y[], num[] are all created as {var name = new createArray(31);} and the resources array is declared to 10 memory spaces because the RTM consists of 10 main steps. The parameter arrays are the same with the other manufacturing processes because the same databases are used.
2. HTML coding has 10 main steps. The forms, $ptimenon_i$, $ptimerec_i$, $ptime_i$, $ltimenon_i$, $ltimerec_i$, $ltime_i$, $etimenon_i$, $etimerec_i$, $etime_i$, $stepcost_i$, $percent_i$ go from 1 to 11 whereas the other forms remain the same.
3. The abbreviations for the main step variables are ts (tool setup), mls (material setup), ml (material loading), cm (close mold), srim (setup resin injection machine), ic (injection/cure), dem (demold), pc (post cure), fn (finishing), in (inspection), and ua (user additional steps).

Chapter 6 Concluding Remarks

6.1 Summary

Because there are neither time nor cost estimation methods or software available to the public for advanced composites (whereas there are many for metal processing), the Cost Estimation Models (CEMs) have been developed. The aim of CEMs is to assist the advanced composites designer in evaluating cost reduction strategies so that s/he can confidently make decisions early during the design phase using information obtained from the models.

A few of the benefits of the CEMs are to assist in the selection of the composites manufacturing process, to provide the process time and cost at different production volumes, to list out the default process plan, and to make databases available for everyone. Essentially, CEMs will enable the designer to estimate and make decisions without data.

The CEM is a process-based cost estimation model that uses the first order dynamic law. The time constant, extensive velocity, equations and other parameters have been based upon the work of [Neoh]. Because the indirect labor costs are generally pooled and distributed to a part on the basis of direct labor hours, using indirect multipliers, only the direct manufacturing cost has been estimated in our CEMs. Fourteen shapes and five processes (ATP, Forming, Hand Lay-up, Pultrusion and RTM) are modeled. The product selection matrix gives guidance to the user by making 'common', 'possible', and 'not common' classifications. 'Common" and 'possible' are treated in detail.

In order to make the CEMs readily available to the public, the Internet technology has been used as the basic tool. JavaScript, HTML and XML have been selected as the programming languages. All model calculations use JavaScript which has been embedded into the HTML interface. Unlike many other languages which perform the necessary executions at the server, JavaScript processes the information at the user's own computer. Additionally, XML databases have been implemented to store the parameters for the process, materials and resources. Some of the many useful tools for the CEM are the display of databases, the individual process parameter HTML page, the time breakdown table, and the quantity classification to part per setup as well as the number of setups.

An example of the Hand Layup of a simple curved part is presented to give a detailed explanation of the mechanics of the CEM. Since all the processes follow the same exact structure, it is easy to apply the concepts from the provided example to other CEMs. For the purpose of future updating and improvements, the coding for all the programming languages is discussed.

For the purpose of being as user friendly as possible, the CEMs were both constructed and tested with satisfactory results.

A brief review of the highlights or main points of each chapter follows:

- Chapter One gives an overview of the research. A brief discussion of advanced composites (characteristics and benefits) is discussed. The five composites manufacturing processes used in the CEMs (ATP, Forming, Hand Layup, Pultrusion, and RTM) are introduced. Reference to the results of the studies by [Neoh] on systems modeling with the first order dynamic is made and the concepts behind the modeling of both computerized and cost estimation models are also presented. Lastly, the computer languages (JavaScript, HTML and XML) used in the CEMs are summarized.
- Chapter Two discusses the model structure and the objectives of the CEMs. After introducing the shapes and processes, which are used throughout the models, a big picture of the CEMs is drawn out and the equations used throughout the CEMs are listed.
- Chapter Three illustrates in a detailed manner how the CEMs work. The Hand Layup of a simple curved part is used as an example. Because all the processes follow the same structure, the reader can easily apply the concepts found in this chapter to all the other processes.
- Chapter Four explains the coding used throughout the CEM and is primarily intended for the programmer in case any maintenance or updating is needed.

- Chapter Five points out differences in other CEMs from the example of the Hand Layup process. The issues, equations, and differences in coding for the other four processes (ATP, Forming, Pultrusion and RTM) are discussed.

6.2 Future Work and Improvements

The goal of this CEM project was to create CEMs that truly work. Nevertheless, once the project is out in the public domain, changes to improve the aesthetics will probably be needed.

At the moment, only material cost, labor cost and machine cost have been included in the model. In the future, many sections may need to be added or updated to the existing models. They include the updating of material database and the adding of tooling, equipment, and assembly sections. Moreover, a systems planning model, using the assistance of the CEMs, will be developed. Finally, the study of the effect of learning curve to the CEMs will be studied because humans learn and get better at their tasks and also because there are economies of scales for large production. With the successful implementation of the inclusion of the tooling section, there is strong confidence of continued use of the CEM structure for years and years to come.

The project hopes to be able to work quite closely with companies in the composites industry in an open exchange of mutually beneficial information. The research group hopes to update the databases with real values from the industry in order to continually improve the CEMs. The companies can more confidently use the CEM framework with their particular parameters for cost estimations. The ultimate goal of this project has been to provide the CEMs to help the composites industry in as many and beneficial ways as possible.

Bibliography

[Beresheim]

G. Beresheim, Part Complexity Based Time Estimation Model for the Automated Tow Placement Process, Internal Report, Laboratory for Manufacturing and Productivity, MIT, January 2001.

[Cincinnati Machine]
N.N.: Cincinnati Machine – FPS Viper 1200. http://www.cinmach.com/products/ adva_set.htm, 11.1.2001

[El Wakil]
S.D. El Wakil, Processes and Design for Manufacturing, 2nd Edition, PWS Publishing Company, Boston, MA, 1998.

[Graham]

I.S. Graham, HTML Sourcebook, John Wiley & Sons, Inc., New York, NY, 1995.

[Gutowski]
T.G. Gutowski, Advanced Composites Manufacturing, John Wiley & Sons, Inc., New York, NY, 1997.

[Haffner, 1]
MIT Composites Group, Laboratory for Manufacturing and Productivity, MIT, Process Description, http://web.mit.edu/lmp/www/composites/costmodel/, 2000.

[Haffner, 2]
S.M Haffner and T.G. Gutowski, Manufacturing Time Estimation Laws for Composite Materials, NSF Conference Report, 1999.

[Haffner, 3]
MIT Composites Group, Laboratory for Manufacturing and Productivity, MIT, Part Description, http://web.mit.edu/lmp/www/composites/costmodel/, 2000.

[Haffner, 4]
S.M Haffner and T.G. Gutowski, Cost Estimation of Composite Parts Example: Pultrusion of a Straight C-Profile, LMP Internal Report, 2000, pp.17-22.

[Haffner, 5]
S.M Haffner and T.G. Gutowski, Cost Estimation of Composite Parts: Hand Layup, LMP Internal Report, 1999, pp.6-9.

[Haffner, 6]
S.M Haffner and T.G. Gutowski, Cost Estimation of Composite Parts Example: ATP of a Small/Large Simply Curved Part, Internal Report, 2000, pp.7-18.

[Haffner, 7]
S.M Haffner and T.G. Gutowski, Cost Estimation of Composite Parts Example:RTM of a Curved L-Profile, LMP Internal Report, 2001, pp.8-14.

[Krolewski]
S.M. Krolewski, An Economics Based Model to Evaluate Advanced Composite Fabrication Technologies, Ph.D. Thesis, Department of Mechanical Engineering, MIT, February 1989.

[Mathieu]
R.G. Mathieu, Manufacturing and the Internet, Engineering and Management Press, Norcross, GA, 1996.

[Nielsen]
J. Nielsen, Designing Web Usability, New Riders Publishing, Indianapolis, IN, 1999.

[Neoh]
E.T. Neoh, Adaptive Framework for Estimating Fabrication Time, Ph.D. Thesis, Department of Mechanical Engineering, MIT, August 1995.

[Norman]
D.A. Norman and S.W. Draper, User Centered System Design, Erlbaum, Hillsdale, NJ, 1986.

[Noton]
B.R Noton, General Use Considerations, Engineered Materials Handbook, ASM International, 1988, Vol.1, pp.35-37.

[Pardi]
W.J. Pardi, XML in Action, Microsoft Press, Redmond, WA, 1999.

[Reinhart]
T.J. Reinhart and L.L. Clements, Introduction to Composites, Engineered Materials Handbook, ASM International, 1988, Vol.1, pp.27-34.

[Schwartz]
M.M. Schwartz, Composite Materials Handbook, 2nd Edition, McGraw-Hill, Inc., 1992.

[Swift]
K.G.Swift and J.D. Booker, Process Selection From Design to Manufacturing, John Wiley & Sons, Inc., New York, NY, 1997.

[Wagner]
R. Wagner, et al., JavaScript Unleashed, Sams.net Publishing, Indianapolis, IN, 1997.